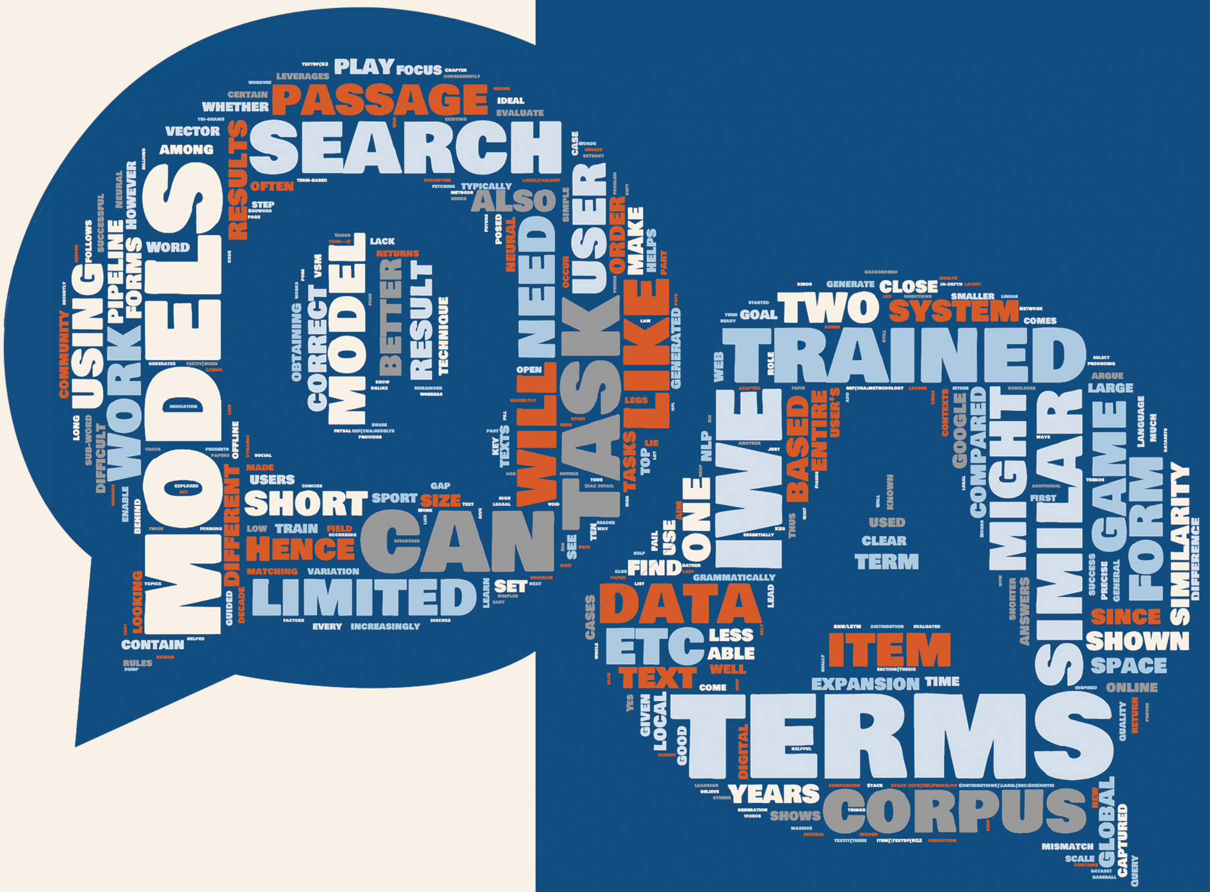
MASTER'S
THESIS



Word Embedding Models for Query Expansion in Answer Passage Retrieval

NIRMAL ROY

Word Embedding Models for Query Expansion in Answer Passage Retrieval

THESIS

submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE

in

COMPUTER SCIENCE

by

NIRMAL ROY
born in Kolkata, India
Student id: 4724429
Email: nroy000@gmail.com

Web Information Systems
Department of Software Technology
Faculty EEMCS, Delft University of Technology
Delft, the Netherlands
http://wis.ewi.tudelft.nl

Abstract

With the increasing popularity of mobile and voice-assisted, extracting short and precise answer passages to open-domain questions is becoming an increasingly important information retrieval (IR) task. The recently released large-scale corpus for answer passage retrieval—`WikiPassageQA` [24]—was shown to be challenging for both traditional retrieval models and neural architectures. One of the classic approaches to improving retrieval effectiveness across tasks is automatic query expansion (QE). QE is the process of reformulating a user's query by adding more terms with the goal of retrieving more relevant information. Word embeddings are commonly employed to obtain QE terms by taking advantage of the low dimensional semantic space formed by these embeddings.

Recently, Diaz et al. [37] showed that QE using word embeddings trained on a *local* query-specific corpus performed better than embeddings that were trained on an entire *global* corpus for document ranking tasks. We aim to examine the effectiveness of QE, specifically using locally-trained word embeddings, in this new context of answer passage retrieval. Additionally, a query-specific corpus can be small in size with limited vocabulary which forms a challenge for training word embedding models. Since the extent to which limited vocabulary influences the semantic information captured by word embeddings is relatively unexplored, we compare two word embedding models—CBOW and IWE—in this thesis. Having the same underlying training philosophy, the IWE model differs from CBOW in two aspects—it incorporates subword information of words and uses a convolutional neural network to learn context representation.

Our results corroborate the findings of Diaz et al. [37]—query-specific data is also beneficial in the task of retrieving passages to open-domain questions. Word embeddings trained on a global corpus fail to capture the nuances of query-specific language present in the answer passages. We also found out that IWE word embeddings capture more semantic information than CBOW word embeddings when trained on local data with a limited vocabulary. Our experiments show that both the IWE model components contribute to the improved quality of word embeddings and consequently better QE terms. Our work can be extended by using the same methodology in other domains or by using different word embedding models to obtain QE terms. The insights from our thesis can help researchers to make an informed decision while choosing word embedding models and training data for their IR and natural language understanding tasks.

Preface

This is going to be one of the longest prefaces for a master's thesis. For a person who stayed with his family in his hometown in India for 23 years, coming to a new country, a new continent was supposed to be an ordeal, full of shocks, cultural and otherwise. And yet, looking back, I have nothing but fond memories from the last two years—memories of how I made great friends and became a better person, both professionally and personally. I have a lot of people to thank for that.

Dr. Claudia Hauff, my supervisor for the first research project of my life—with your critical feedback, throughout my thesis, I saw exponential improvements in the way I write, think about a research problem and tackle it. I thank you sincerely for converting my work mindset from that of an engineer to that of a scientist. I am excited and looking forward to working with you for the next four years of my research career.

Felipe and Gustavo, thank you for bringing me back to reality every time I thought my thesis was the worst piece of research ever. I got the most interesting ideas for my research while brainstorming with you. Most importantly, your feedback and patience while answering my doubts, not only inspired me to push for that extra mile but also to help future master students similarly. I would also like to thank the rest of the members from the lab—Arthur, Wanning, Alex and specially Daan and Kun, for being my research 'buddies' during the first and last phases of my thesis respectively.

People who know me are familiar with how much I love football—one of the most fulfilling things from these years was the fact that I got to play to my heart's content (and became a better player in the process). I would like to thank my team—Mo, Slawek, Roel, Keshav, Balja, Reinaldo and the rest. Playing with them was the welcoming break that helped me keep motivated during my studies. The Introduction Program team—Sophie, Marjo and Rianne, thank you for allowing me to work with you and the rest—Andrea, Sebastian, Kostas, Wissam, Tugba, Santiago, Sahana, Mahtab for making it one of the most fun experiences.

Francisco (my first friend in Delft), Daniel, Ania, Ombretta, Lun, Johan, Sanny, Pradeep—could have never asked for a better set of friends for my master's life. I wish to carry forward our friendship through the next phases of our lives. Karol—best gym partner, fellow spice, sports and skate enthusiast, kindest Polish guy ever—thanks for extending your helping hand whenever I need it. Barbara—best project partner, supervisor, and friend—graduation has been a long journey for both of us and I owe a lot of my gathered skills and acumen to you. I found invaluable friendship, far from home, in both of you. Martin—I could not have asked for a better flatmateturned-brother. I am wiser, fitter, happier and a better researcher because you helped me burst my bubble with our conversations. My Indian contingent—Dhruv, Kanav, Sumi, Arka, Tavishi, Gou, Sanjeev, Rahul, Bhati, Sasha—I never really felt away from home because of you. I will always cherish every single moment—the food, drinks, travels, football and incessant jokes—which forms the majority of my happy and warm memories of the last two years.

Prasid, Sourav, Arpan, Deeptish—you have been instrumental in my decision of doing a masters abroad and also in shaping many of my good (and bad) virtues which I still carry. Your support has been incessant and I cannot thank you enough for it. Words are not enough to express how my family and parents have helped me on this journey. Your sacrifice and love has always been and always will be my backbone. Making you proud drives me to be successful and more importantly, to be a better person in life. Lastly, Meghdipa—none of this would mean much without you by my side. The last two years have been a roller coaster for both of us but whenever I pined for home, I have found it in you. I hope to have you with me for all my future endeavors and hope to be with you in yours.

The destination, finishing my master's degree, gives me happiness and relief but the journey, with all of you, is going to be my keepsake.

NIRMAL ROY
Delft, the Netherlands
July 31, 2019

Contents

List of Figures

List of Tables

Symbols

$\mathbf{d}$	document
$\mathbf{D}$	set/collection of documents
$\mathbf{q}$	query
$\mathbf{q}_i$	i-th term in a query
$\mathbf{Q}$	set of queries
w	word/term
S	sentece
$tf(t,\mathbf{d})$	term frequency of t in document $\mathbf{d}$
$idf(t)$	inverse document frequency of term t
$S(\mathbf{d},\mathbf{q})$	Relevance score assigned to document $\mathbf{d}$ with respect to query $\mathbf{q}$ by retrieval scoring function S
$\lvert\mathbf{d}\rvert$	number of words in document $\mathbf{d}$
$\lvert\mathbf{q}\rvert$	number of terms in query $\mathbf{q}$
$\mathbf{V}$	vocabulary
$\lvert\mathbf{V}\rvert$	number of words in the vocabulary/ size of the vocabulary
$\mathcal{R}$	relevance model—probability distribution of words in relevant documents
$\mathcal{D}$	document language model—probability distribution of words in one document
$\mathbf{U}$	embedding matrix
$\mathbf{w}$	embedding vector of word w
$\lvert\mathbf{V}_r\rvert$	size of word root vocabulary
$\lvert\mathbf{V}_t\rvert$	size of character tri-gram vocabulary
C	number of context words
N^-	number of negatively sampled words
α	learning rate of word embedding models
d	dimensions of word embeddings
k	number of expansion terms
λ	interpolation weight for the expanded language model

Chapter 1

Introduction

Retrieving information from search engines (Google[1], Bing[2] etc.), social question answering (QA) sites (Yahoo Answers[3], Stack Overflow[4] etc.), exploratory searching (travel planning, personal health research etc.) forms an important part of our lives in the present world. Humans interact with these systems, either through desktops or mobile devices, to find information and facts which in turn helps them to make informed decisions. The interaction is done in the form of a query in order to communicate their information need to the IR systems. The objective of an information retrieval (IR) system is to satisfy this information need by retrieving a list of documents, passages, sentences, etc. from a collection where the relevant results are usually kept at the top. The retrieved information can be in other modality as well - images, videos or audio which has become popular in the last few decades [30]. In this thesis, we only consider **text retrieval** where both the queries and the retrieved information (search results) are in the form of natural language.

When the query accurately describes the information a user is seeking for, the system is likely to return good results, given the relevant information is present in the collection of the IR system. In reality, however, the query submitted by a user might form a poor representation of the underlying information need [132]. For example, a user looking to understand how the game of football is played might formulate their web-search query simply as 'football'. These short queries can be ambiguous as well—it is not clear from the query whether the user is looking for the latest football news or any specific information regarding the sport. Fig. 1.1 shows the result returned by Google in response to the above query. Since the information need was not clear from the query formulation, the search system failed to retrieve results that are relevant to the user. Furthermore, as famously conceptualized by Nicholas Belkin in his *Anomalous State of Knowledge* [6], a user often does not know what their actual information need is. **Query refinement techniques** [69, 4] aim to minimize this gap between the actual information need of the user and their formulated query. **Query expansion** (QE) is one such technique among others like spelling correction, query auto-completion. QE, as the name suggests, is the technique of *expanding* a user's original query with new terms that are semantically similar to the query and relevant to the information need

[1]See https://www.google.com/

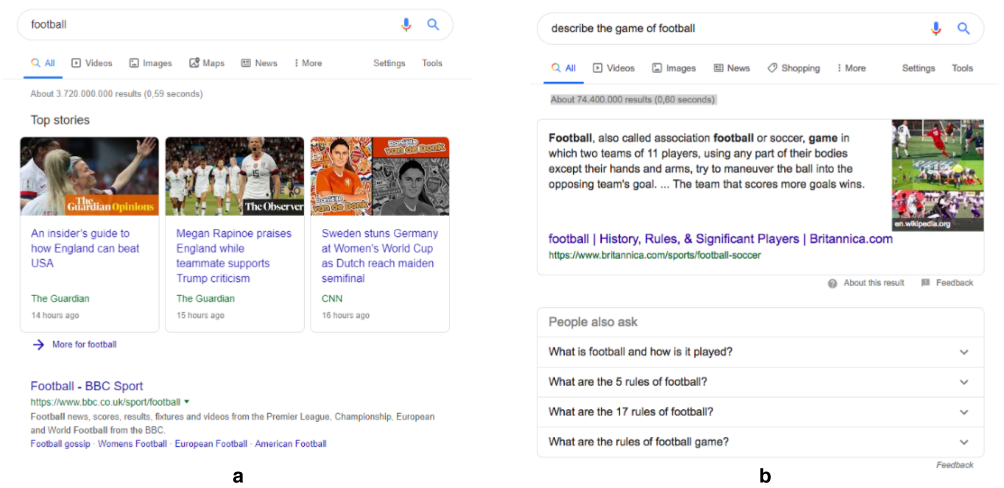
[2]See https://www.bing.com/.

[3]See https://answers.yahoo.com/.

[4]See https://stackoverflow.com/.

of the user [38, 14]. For example, a good QE technique will add the terms 'game', 'rules', 'play' etc. to the query mentioned above.

Over the years, a number of approaches [116, 28] have been proposed for QE which we have elaborated more in Section 2.3. Currently, continuous space **word embedding** models[5] like `word2vec` [87], `GloVe` [105] etc. are popularly employed to obtain QE terms [37, 118, 68, 100, 61]. These models project words in a vocabulary to a dense, lower-dimensional continuous space where each word is represented as a point. These numerical vector representations of words have the ability to model word similarity and other semantic relationships. Words having similar semantic meaning ('football', 'game', 'play' etc.) usually lie close to each other in the embedding space. Research in natural language processing (NLP) [85, 72, 105, 77] community have shown the effectiveness of word embeddings in analogy and word similarity tasks. This attribute of term relatedness can also be exploited to find QE terms to a given query. We will be able to obtain 'game' and 'play' as QE terms by searching the embedding space in the vicinity of the query term 'football'. QE using word embeddings have been shown to improve the retrieval effectiveness [2, 37, 68] of term-matching based traditional IR models[6] (eg. BM25, Query Likelihood) in the **ad-hoc search** task - the task of ranking documents in response to short queries.



About 3.720.000.000 results (0,59 seconds)

Top stories

An insider's guide to how England can beat USA
The Guardian
14 hours ago

Megan Rapinoe praises England while teammate supports Trump criticism
The Guardian
15 hours ago

Sweden stuns Germany at Women's World Cup as Dutch reach maiden semifinal
CNN
16 hours ago

More for football

Football - BBC Sport
https://www.bbc.co.uk/sport/football
Football news, scores, results, fixtures and videos from the Premier League, Championship, European and World Football from the BBC.
Football gossip · Womens Football · European Football · American Football

a

About 74.400.000 results (0,60 seconds)

Football, also called association **football** or soccer, **game** in which two teams of 11 players, using any part of their bodies except their hands and arms, try to maneuver the ball into the opposing team's goal. ... The team that scores more goals wins.

en.wikipedia.org

football | History, Rules, & Significant Players | Britannica.com
https://www.britannica.com/sports/football-soccer

People also ask

What is football and how is it played?

What are the 5 rules of football?

What are the 17 rules of football?

What are the rules of football game?

b

Figure 1.1: Google search result in response to the keyword query (a) 'football' and grammatically correct question (b) 'describe the game of football'.

In our work, we evaluate the effectiveness of QE in a context that is relatively novel in the field of IR: **open-domain QA**. It aims to answer queries posed by users in natural language [19, 134, 24] with grammatically correct syntax—'Describe the game of football'. As we can see, in contrast to the keyword query 'football', this is more well-defined and less ambiguous—the information need is clear. Fig. 1.1 shows the response to the natural language query—it retrieved the information relevant to the user. However, natural language questions come with their own complexities. The IR system needs to understand the semantic relationship between various query terms to

[5]Word embedding models are introduced in details in Section 2.2

[6]Traditional IR models are introduced in Section 2.1.2

retrieve the relevant and correct information[7]. Moreover, various terms in the query might not be present in the relevant answer passage or document. For example, in the answer retrieved in Fig. 1.1, the word 'Describe' is not present. This is what is typically known as *vocabulary mismatch* problem [161, 125] where a simple term matching based IR system might fail to retrieve relevant information because of the absence of important query terms. Without the keyword 'describe' the intent of the query ('the game of football') might change as it is unclear now whether the user is looking for how to play the sport or a PC/video game about the sport. Hence, QE can reduce this gap by expanding the query with semantically similar words to describe what will be present in relevant answer passages.

QA is a challenging and one of the earliest tasks in Natural Language Processing (NLP) and has been a subject of research since 1960s [130, 65, 17, 67]. *Open domain* QA is not restricted to any specific domain or topic [19, 146]. Whereas in *closed domain* QA question topics come from one specific domain like medicine, law, tourism, etc. Open-domain QA forms an important step towards the QA ideal, which is a general question answering system that is capable of answering a question from any domain posed by an user [65, 17, 62]. In IR community, open-domain QA has received much attention since 1999, when the QA track was first introduced in Text Retrieval Conference (TREC) competitions by the National Institute of Standards and Technology (NIST) [144, 17, 89]. This was followed by development of a number of QA IR models [44, 111, 26, 146, 155] over the years. They usually contain two components - (i) using an IR model to select a candidate set of passages/documents from open-domain knowledge sources like Wikiepdia[8] or a subset of World Wide Web pages etc and (ii) selecting the correct answer from the candidate or re-ranking them to keep the answer passages/documents at the top. The first component of selecting candidate answer texts is usually performed by one of the terms based IR models as they are fast and robust. The more recent neural network-based IR models [23, 153, 18, 112, 155] focus on the second component of re-ranking the answers with the goal of increased retrieval effectiveness.

1.1 Research Motivation

With the increasing popularity of digital assistants like Google Assistant[9], Siri[10], Cortana[11], and Alexa[12] open-domain QA is becoming an increasingly important retrieval task. A user interacting with these digital assistants would prefer concise answers instead of entire documents. Hence, we focus on the task of **passage retrieval** to open-domain QA tasks in our thesis. Passages are shorter pieces of text than documents. While ranking of documents is typically performed for the ad-hoc search task,

[7]Apparently easy questions can be difficult for search systems. For example, a Google search on the question 'How many legs does a horse have' returns 'four' as an answer. However, 'Number of legs horses have' returns 'six'.

[8]See https://en.m.wikipedia.org/wiki/Wikipedia.

[9]See https://assistant.google.com/.

[10]See https://www.apple.com/ios/siri/.

[11]See https://www.microsoft.com/en-us/windows/cortana.

[12]See https://alexa.amazon.com.

retrieving short and precise pieces of answer text like passages forms one of the key aspects of open-domain QA systems [158, 122, 24, 33].

Although existing research [37, 118, 2, 68] has shown the effectiveness of QE using word embeddings in the ad-hoc search task, *there is lack of literature when it comes to analyzing the same for the task of passage retrieval for open-domain QA*. Specifically, there are two factors that can make the generation of QE terms difficult in our open-domain QA task in comparison to the ad-hoc search task—(i) the QE terms need to be semantically similar to all the important question terms ('describe', 'game', 'football' etc.) which can be challenging when the questions are long and (ii) the QE terms have to retrieve passages which are not only smaller but also less topically diverse than documents of the ad-hoc search task [24, 141].

In this work, we aim to re-implement the retrieval pipeline recently proposed by Diaz et al. [37] and evaluated on the ad-hoc search task. The key idea behind the pipeline is to reap the benefits of so-called *locally trained embeddings* (derived from a query-specific corpus) over *globally trained embeddings* (derived from an entire corpus) for QE. The unigram distribution of text on subtopics of a collection differs from that of the whole corpora [29]. For example, the language will be different in the documents pertaining to football than politics. The word embedding models essentially capture word co-occurrence information—words occurring in the same context are semantically similar [55, 42] and thus lie close to each other in the embedding space [85]. As a result, the trained word embeddings will also be different when they are trained on documents specific to a topic or a query. Table. 1.1 shows the difference in similar terms produced by a word embedding model when trained on a query-specific corpus compared to when trained on a global corpus. In this case, we see that terms specific to 'football' are obtained by the locally-trained word embeddings, whereas more global relationships are captured by the globally-trained embeddings.

Table 1.1: Terms similar to 'football' for a word embedding model trained on a general corpus and another trained only on documents related to 'football'. Figure adapted from [37].

Global	**Local**
soccer	players
baseball	game
basketball	role
league	pitch
rugby	striker
hockey	international
club	federation
player	goal
footballer	futsal
nfl	keeper

The candidate set of passages to each question forms the query-specific corpora for training the word embedding models for our experiments. Hence, our task forms the ideal setting to evaluate the reproducibility of the approach proposed by Diaz et al.

[37] in this new domain. Furthermore, passages are topically constrained pieces of text and as pointed out in [37], global embeddings fail to capture the nuances of topic-specific language. We hypothesize that *it is better to train our word embedding models on query-specific data instead of a global corpus one for obtaining QE terms for our task of passage retrieval to open domain QA* (**H1**).

Word embedding models are usually trained on data with vocabulary (number of unique words) size of more than 50k [85, 13]. Passages being shorter pieces of text, we would need to retrieve a large number of candidate passages in order to have a similar vocabulary size. This would result in a larger number of non-relevant passages among the candidates and thereby defeating the purpose of query-specific data. Ideally, we should retrieve a limited number of candidate passages to train our local embedding model. However, the vocabulary size for training our word embedding models will be much smaller as well. *There has been no detailed investigation previously on the outcomes when word embedding models are trained on limited data—whether the generated word embeddings still capture semantic relationship or whether limited data render them completely useless*. Obtaining and analyzing QE terms generated by the word embeddings will enable us to investigate their quality when they are trained on limited data. The QE terms are a good indication of the semantic information captured in the embeddings. If the quality of the embeddings is low, it would result in QE terms having little or no semantic similarity with the query terms.

In our thesis, we use two word embedding models—CBOW [85] and IWE [13]—for obtaining the QE terms in our task of passage retrieval[13]. Both these models are trained using the same objective: prediction of a word based on the words in its context. The difference lies in how the models form a representation of the context. CBOW simply concatenates the context words without any word order information to form the context representation. On the other hand, IWE uses a convolutional neural network (CNN) [43, 71] to learn structural information (word order) about the context and form the context representation. Moreover, IWE leverages sub-word information (like character trigrams) of the words while forming the context representations. We hypothesize that *when the models are locally trained on candidate set of passages having a limited vocabulary, IWE word embeddings will capture better semantic relationships and consequently generate better QE terms for passage retrieval than the CBOW model* (**H2**).

Thus the work in our thesis is guided by the following research questions:

RQ1 Does QE based on locally-trained word embeddings [37] improve the effectiveness of traditional term-based retrieval model (Query Likelihood) for the task of answer passage retrieval compared to globally-trained embeddings?

RQ2 Does QE based on IWE word embeddings improve the effectiveness of traditional retrieval models compared to CBOW word embeddings when the models are locally-trained on limited data?

[13]The models are introduced in details in Chapter 3

1.2 Scientific Contributions

The main contribution of our work can be summarized as follows:

- We have explored QE using word embeddings in the task of answer passage retrieval for open-domain QA. Specifically, we have compared the retrieval effectiveness after QE using both CBOW and IWE word embedding models trained *globally* and *locally*.
- We have compared the performance of CBOW and IWE word embedding models with the help of QE terms these two models generate. The difference in QE terms generated by the two models and their retrieval effectiveness have shown that IWE word embeddings capture more semantic information than CBOW word embeddings when trained on limited data. We have shown evidence that the incorporation of sub-word information and convolution feature learning of context helps in the same.
- We have conducted extensive error analysis on cases when QE using CBOW or IWE word embeddings failed to increase the retrieval effectiveness of traditional IR models. The analysis has led to the identification of certain shortcomings of the word embedding models and the adopted retrieval pipeline.
- We have implemented IWE word embedding model [13] from scratch in PyTorch[14]. We have also made code of our IWE implementation[15] and experiments[16] publicly available.

1.3 Thesis Outline

The remainder of our thesis is structured as follows:

- **Chapter 2: Related Work** provides the reader with necessary background information of IR and discuss research related to our work.
- **Chapter 3: Methodology** describes all the algorithms employed in our work in detail and discusses our approach for answering the research questions.
- **Chapter 4: Experiments and Results** presents the findings of the experiments designed to answer our research questions together with an in-depth analysis of the performance of the word embedding models.
- **Chapter 5: Conclusion and Future Works** concludes and summarizes our findings and analysis. It also puts forward several possible research directions.

[14] An open source deep learning library for Python. See https://pytorch.org/.
[15] See https://github.com/roynirmal/IWE.
[16] See https://github.com/roynirmal/queryExpWikiPassageQA.

Chapter 2

Related Works

In this chapter, we first provide background information to make the readers familiar to the concepts of our research topics - namely **IR**, (Section 2.1), **word embeddings** (Section 2.2) and **QE** (Section 2.3).

Secondly, we provide a brief literature survey of research, discussing their merits and limitations, that are related to our work. We identified two such areas of work: word embeddings in IR - the neural IR paradigm (included in section 2.2), and query expansion using word embeddings (Section 2.4).

2.1 Information Retrieval

The term IR can mean a large number of things. Taking a notebook out of your bag to read a certain topic is also a form of IR. According to Manning et al. [82],

> Information retrieval (IR) is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from within large collections (usually stored on computers).

Data that cannot be maintained in relational databases are typically unstructured. Text data, which is the focus of our thesis, however, is never truly unstructured—they can be divided into paragraphs, titles, footnote, etc. Text retrieval is the task of finding relevant information from one or more such parts of text data in response to a user query. The information can be presented to the user in a number of ways depending on the task and IR system. For example, in personalized search in desktops or e-mails, users are typically looking for one relevant document/text/mail. Ranking of documents, with the relevant ones at the top, is a central task in IR research [90]. The most common example of this IR task in our daily lives is a typical web search engine like Google, Yahoo, Bing, etc. In the following sections, we will first discuss two important IR tasks that are directly relevant to our thesis, followed by a discussion on traditional IR models and finally the topic of evaluation of an IR system.

2.1.1 IR Tasks

The text retrieval research can be subdivided into a number of tasks such as ad-hoc retrieval, query understanding, question answering or more novel tasks like complex

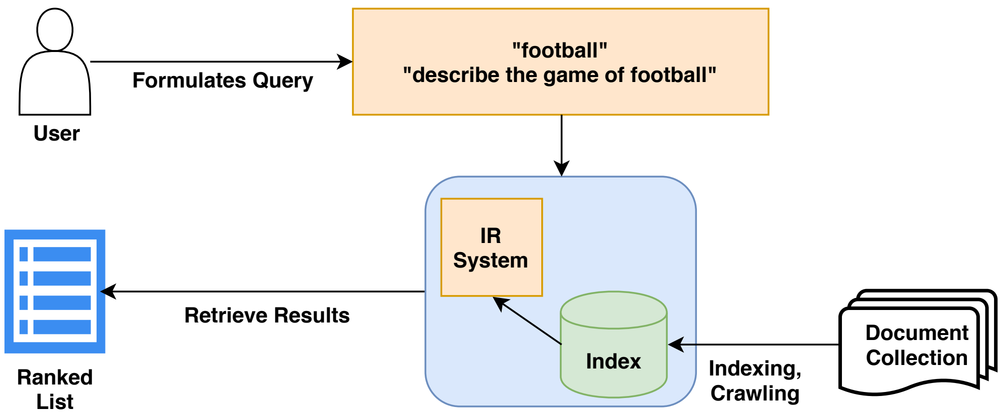


Figure 2.1: A pictorial representation of a typical IR process

question answering [97] depending on the *target textual unit* (TTU) or TTU pairs they are dealing with. This thesis focuses on ad-hoc retrieval and question answering which are elaborated in the following segment.

Ad-hoc retrieval is the IR task of retrieving a ranked list of documents from a large collection in response to a user's query [138]. It is the main task of popular ad-hoc and Web tracks of the Text REtrieval Conference (TREC) [143] and performed by search engines like Google, Bing etc [92]. For this task, the TTU pair is query and document. Although traditionally ad-hoc retrieval was used for obtaining news reports or government documents by librarians or information experts [3], since early 1990s web search is the most common example. In an IR system, the number of documents in the collection remains relatively static although the number of possible queries is unlimited - hence the term *ad-hoc* [3]. The documents in the collection are indexed to facilitate their search among millions of other documents. The length of the query submitted by the user can vary from a few terms to a few sentences, while in web search it is typically in the form of short keywords [83]. Hence, the query might lead to an information need that is *ill-defined*. This can lead to queries that are ambiguous, as discussed in Chapter 1. Given a query, the goal of the IR model in this task is to rank relevant documents higher than the non-relevant documents in the collection.

Open domain QA is the task of retrieving comprehensible answer texts to user queries that are in the form of natural language questions. Naturally, the TTU pair for this task is a question and answer. QA can further be of two types depending on the length of text necessary to answer a user's question [136, 153, 24]. It is called **factoid QA** when the length of the answer varies from a few words to a few phrases - "Who won the football world cup in 2018?". The length of answers to **non-factoid QA** lies between a sentence or a phrase (factoid QA) and an entire document - 'Describe the game of football'. In our thesis, we focus on non-factoid questions where our task is to retrieve a ranked list of answer passages. The queries in QA are *well-defined* in comparison to ad-hoc retrieval—the information need is typically clearer. However, retrieving short spans of answer text poses a unique challenge in comparison to retrieving documents [92]. Since the short answer spans tend to be on point with respect to a single topic in comparison to a long document, they might use a different language than the questions. Often important question terms are not present in the answer passages. Hence, retrieval systems designed for QA tasks need to focus more on modeling semantic patterns expected in the answer than an *exact matching* of query terms [92]. Fig. 2.1 shows a graphical representation of a typical IR ad-hoc/QA task. Similar to ad-hoc search, the goal of the IR model in our task of open-domain QA to non-factoid questions is to rank relevant passages higher than the non-relevant passages.

2.1.2 IR models

In an ideal scenario, a user would manually look at all documents in a collection and keep the ones she finds relevant to particular information. Of course, this solution is impractical and it necessitates a model to automate this process for her. The goal of an IR model is to retrieve documents that are relevant to the users in response to their query. However, relevance is a complex notion [121] might vary between different users or for the same user in a different time. This poses one of the major challenges for designing retrieval models. For ad-hoc or passage retrieval tasks, a good IR model is one that retrieves all relevant documents/passages at the top of a ranked list. For simplicity, henceforth if not mentioned otherwise, we will refer to both documents (for ad-hoc search) and passages (for our task of passage retrieval) as documents. From a holistic view, the task of an IR model can be simplified into three components:

- Given a query $\mathbf{q}$, calculate a score $S(\mathbf{d},\mathbf{q})$ for every document $\mathbf{d}$ in the collection $\mathbf{D}$ which is a measure of the relevance of the document to $\mathbf{q}$.
- Rank documents in descending order of their score $S(\mathbf{d},\mathbf{q})$.
- Produce the top n results to the user.

Over the course of 5 decades of research, various IR models have been proposed [114, 82]. The models differ in how they score the documents and in their assumptions behind the scoring functions but have the same goal of retrieving the most relevant results to the users. One of the earliest of such approaches is the **boolean retrieval model**, where a $\mathbf{q}$ is represented as a logical combination of words. The model returns a set (instead of a ranked list) of documents that *exactly* matches the query with the assumption that all matching documents have the same relevance. However, this model is rarely used in current IR research involving ad-hoc search task or open-domain QA. This was followed by the **vector space model** (VSM) [120], which we discuss in more detail in Section 2.2.1 as they form an important background to word embeddings. The more recent approaches which are used in modern IR research are **probabilistic models** (PM) like BM25 and language modeling.

PM was developed following decades of empirical work on IR models when it was necessary to have more theoretical evidence and explicit assumptions [82]. Probabilistic IR models measure the relevance of documents based on probabilities estimated as accurately as possible based on the data available to the models [115]. **Binary independence model** (BIM) is a PM where documents are represented as a vector of binary features (1 if a term is present, 0 otherwise) and assume independence of terms. The key idea behind BIM is to rank documents based on the ratio of its probability of being relevant to the query to that of being non-relevant. The assumption of term independence simplifies the calculation of the probability that the document is relevant to the query. BIM is the basis of **BM25** [90] which is not only one of the most popular baselines of current IR research but the first PM model to consistently outperform previous approaches. BM25 extends BIM by incorporating document weights and **term frequency** (TF) $tf(w,\mathbf{d})$ which is the frequency of a word w in a document. The equation of BM25 [41] is presented below.

$$BM25(\mathbf{d},\mathbf{q}) = \sum_{\mathbf{q}_i \in \mathbf{q}} idf(\mathbf{q}_i) . \frac{tf(\mathbf{q}_i,\mathbf{d}).(k_1+1)}{tf(\mathbf{q}_i,\mathbf{d}) + k_1.(1-b+b\frac{|\mathbf{d}|}{avgdl})} . \frac{(k_2+1)tf(t_q,\mathbf{d})}{k_2+tf(\mathbf{q}_i,\mathbf{d})} . \tag{2.1}$$

where k_1, k_2 and b are parameters of BM25 that need to be tuned and $avgdl$ is the average length of documents in the collection $\mathbf{D}$. $idf(w)$ is known as **inverse document frequency** (IDF) of a word and measures the importance of a term based on number of documents in $\mathbf{D}$ that contains the term (also known as document frequency of the term $df(w)$. IDF is calculated as the following [41, 92]

$$idf(w) = \log \frac{|\mathbf{D}| - df(w) + 0.5}{df(w) + 0.5} \tag{2.2}$$

Language modeling [109] is having a probability distribution over the vocabulary $\mathbf{V}$ (number of unique words/stems/phrases) in a collection or document. While BM25 uses various query term statistics to directly compute the relevance of a document with respect to the query, the main intuition of retrieval using language modeling is to estimate the *likelihood* $P(\mathbf{q}|\mathbf{d})$ of generating query terms if words are randomly sampled from a document [82]. Under the assumption that the query terms are independent of each other computing the likelihood becomes easier,

$$P(\mathbf{q}|\mathbf{d}) = \prod_{\mathbf{q}_i \in \mathbf{q}} P(\mathbf{q}_i|\mathbf{d}). \tag{2.3}$$

The documents are then ranked based on the relevance score which is the product of the likelihood estimates of all query terms (using Bayes Theorem). This model is also referred to as **query likelihood** (QL). The prior probabilities of documents can assumed to be uniform or can be estimated from training data based on document length, source etc. [82]. Often, to avoid the problem of having 0 probability (and consequently 0 relevance score for a document), QL use *smoothing* [81, 82] by sampling query terms from the collection if they are unseen in the documents. For our thesis, we use a QL model with Dirichlet smoothing [81] as our retrieval model since this was also used as a baseline model by Diaz et al. [37]. The formula [41] for the same is presented below. Here μ is the smoothing parameter and $|\mathbf{d}|$ is the document length or the number of words in the document.

$$QL(\mathbf{d},\mathbf{q}) = \prod_{\mathbf{q}_i \in \mathbf{q}} \left(tf(\mathbf{q}_i,\mathbf{d}) + \mu \frac{\sum_{\bar{\mathbf{d}} \in \mathbf{D}} tf(\mathbf{q}_i,\bar{\mathbf{d}})}{\sum_{\bar{\mathbf{d}} \in \mathbf{D}} |\bar{\mathbf{d}}|} \right) / \left(|\mathbf{d}| + \mu \right) \tag{2.4}$$

More recent IR models include translation models, pseudo relevance feedback (PRF) [28], learning to rank (LTR) [52, 51] and the latest neural IR models (neu-IR) [126, 127, 50, 91]. The PRF model employs query expansion and will be explained in more detail in Section 2.3. LTR and neu-IR approaches can be used to extend traditional approaches by re-ranking their results either by using handcrafted features (LTR) or by learning features automatically (neu-IR) to represent input (more discussion in section 2.2.6).

2.1.3 Relevance, Evaluation and Metrics in IR

As discussed in Section 2.1.3, relevance forms a key notion in IR and it is one of the most important factors for evaluating IR models. In conjunction with effectiveness, an IR model should also be efficient in terms of the computer resources used such as memory, storage, and time [114]. A good IR model retrieves all relevant documents and as few non-relevant documents as possible. Relevance is a complex, intuitive concept of human and it can vary between users or the same user in different time [114, 121]. Hence, the goal of evaluation measures in IR tasks like ad-hoc search or passage retrieval to open-domain QA is to assess how well the model meets the information need of the users with the ranked list of documents it produces [142]. The *user-based* evaluations measure user satisfaction in terms of corpus coverage, time lag for retrieval, presentation of output etc [82]. However, these kind of evaluations are not only difficult or expensive but it also raises questions regarding reproducibility and re-usability of these results [114]. In an attempt to overcome these difficulties and to quantify the difference between system performances, IR researchers primarily use an empirical, *test collection* based approach known as batch or offline evaluation. This IR evaluation methodology was developed by Cyril Cleverdon [22] at the College of Aeronautics at Cranfield in the 1960s and is also known as **Cranfield evaluation paradigm**. The methodology consists of three key components:

- a collection of documents **D**,
- a set of topics or queries **Q**,
- a complete set of binary *relevance judgments* or `qrel` for each document annotated by human assessors.

The modern evaluation has adapted the methodology in terms of relevance judgment where a multi-graded or user-dependent decision is used instead of binary judgments. Moreover, depth pooling[1] [131] is commonly used to avoid the cost of labeling *all* documents, whether they are relevant or not, for every topic. The general idea remains the same - simulate user-based evaluations with the help of these cheaper, reusable and reproducible test collections. The only drawback of this methodology is that they might not reflect the gains or satisfaction of real users [82]. To compare the performance of different approaches, we need evaluation metrics together with the test collection. The evaluation metrics used in our thesis are introduced in Section 3.4.1.

2.2 From Words to Vectors

How can machines understand words? Language, in its written, spoken or signed form, has been the medium of knowledge transfer throughout the course of human-

[1]It is used in TREC and NTCIR

ity. To allow machines or computers to use this knowledge and take part in daily interactions with humans, we need to represent language using mathematical models - models that will help machines translate and understand natural language for their own computation. Word embeddings, the representation of words in numerical vectors of real numbers, stems from a long line of research in NLP - a research field focused on developing mathematical and computational models of language.

The design of mathematical models for language can be dated back to Indian and Greek philosophers of 4th Century BC [113]. In the early 17th century, Leibniz and Descartes put forward theoretical proposals which would relate words between different languages - one of the earliest known work of **machine translation** (MT). The research of MT in the early 20th century was mostly theoretical and unsuccessful owing to lack of computing resources - the task of encoding language into computer programs proved to be more complex than what the researchers had imagined [53]. American linguist Noam Chomsky published *Syntactic Structures* [21] in 1957 where he introduced the concept of a generative grammar: rule-based descriptions of syntactic structures. It is believed that most of NLP research since 1957 has been influenced by Chomsky's work [53]. The period of 1960-1970 saw key developments in the research of encoding grammar, syntax and semantics for machine understanding [147, 123] and development of NLP systems like SHRDLU[2], PARRY[3], etc. Most of these systems and methods were based on a complex set of rules and the focus was on executing a specific task.

Introduction of **machine learning** (ML) algorithms for language processing in the 1980s not only brought a new direction in NLP research but also necessitated the representation of language or text in the form of numerical vectors - vectors that encode important syntactic and semantic information. Word embedding[4] is the collective name for a variety of techniques that are involved with the mathematical representation of words in a low-dimensional continuous vectors space. We can broadly divide the approaches into two categories [5, 105]: **count based word embeddings** which employs global corpus statistics like word frequencies and **prediction based word embeddings** which are typically neural models trained by setting up a prediction task where the objective is to predict a word given the words in its context. We believe the development and emergence of the two types of word embeddings can be respectively attributed to VSM and **neural network language models** (NNLM) which we discuss in this chapter.

2.2.1 The Vector Space Model

The technique of representing words as numerical vectors is typically attributed to the development of the VSM, specifically the work of Salton et al. [120] where they used word frequencies in a collection to capture the semantic information of a text. Prior to that, in linguistics, the *distributional hypothesis* [55], aimed at quantifying the semantic similarity of terms based on their distribution in large samples of language data. Firth [42] famously purported this idea by stating "a word is known by the company

[2]See http://hci.stanford.edu/~winograd/shrdlu/.

[3]See https://phrasee.co/parry-the-a-i-chatterbot-from-1972/.

[4]To our knowledge, the term was first used by Morin and Bengio [95]. Models prior to that used to call them *distributed representation of words*.

it keeps" - words appearing in the same context tend to be semantically similar. The common theme that underlines various forms of VSM and the distributional hypothesis can be stated as *statistical semantics* - that is the statistical pattern of human word usage can be leveraged to understand what people mean.

The VSM of Salton et al. [120] represented queries and documents as $|\mathbf{V}|$-dimensional vectors where $|\mathbf{V}|$ is the size of the vocabulary. Each element in these $|\mathbf{V}|$-dimensional vectors corresponds to a word and can be either binary (1 if the word is present in the document, 0 otherwise) or real numbers. Commonly, the elements are weighted by their TF-IDF (introduced in Section 2.1.2) to capture how informative each word is. Depending on the vocabulary, $|\mathbf{V}|$ can be in millions - leading to a very high dimensional vector space. Given such vector representations, we can rank documents based on their similarity to a query. The most popularly used similarity score is cosine similarity (see equation 2.5) which we employ for various tasks in our thesis as well.

$$Cosine(\mathbf{d}_i, \mathbf{q}) = \frac{\sum_{j=1}^{|\mathbf{V}|} \mathbf{d}_{ij} \cdot q_j}{\sqrt{\sum_{j=1}^{|\mathbf{V}|} \mathbf{d}_{ij}^2 \cdot \sum_{j=1}^{|\mathbf{V}|} q_j^2}} \tag{2.5}$$

where $\mathbf{d}_i = (\mathbf{d}_{i1}, \mathbf{d}_{i2}, ..., \mathbf{d}_{i|\mathbf{V}|})$ and $\mathbf{q} = (q_1, q_2, ..., q_{|\mathbf{V}|})$ are the representation of document $\mathbf{d}_i$ and query $\mathbf{q}$ in this vector space, respectively. This representation is also known as the *term-document* representation [139] where the document vector represents the documents as bag of words in each column. In IR, the *bag of words hypothesis* states that frequency of words in a document tends to indicate the relevance of a document to a query [120, 139]. The cosine similarities in Equation 2.5 essentially captures this notion of relevance.

Deerwester et al. [32] pointed out that we can also compute word similarities by considering the rows of the above term-document representation instead of the columns. This shifted the focus towards representing *words* in a numerical vector in various ways. The words can be represented by a binary vector or one-hot representation where the size of the vector is the size of the vocabulary. For example, in a vocabulary of 10,000 words if 'football' is the third word it will be represented by the 10,000-dimensional vector $[0, 0, 1, 0...0]$. The words can also be represented by various hand-crafted features like documents containing the word [32], grammatical dependencies [76] character tri-graphs etc. Lund and Burgess [78] introduced the *Hyperspace Analogue to Language* (HAL) model where each word was represented by its co-occurrence with neighboring or context words. In this model, stop-words like *the*, contributed disproportionately, since they did not apply any normalization to the co-occurrence counts. Rohde et al. [117] fixed the issue in their *COALS* model by considering conditional co-occurrence - a normalization technique by calculating the ratio of the likelihood of co-occurrence of word A with word B to that with a random word from the vocabulary. In general, we can consider these representations as a word-context matrix where the context is given by words, phrases, sentences, documents or other features as mentioned above [139].

2.2.2 Count Based Word Embeddings

The vector space representation can highlight the semantic relationship between words but they suffer from a few drawbacks. Since the dimensions are in the order of the vocabulary of the underlying corpus or the number of documents, they can be extremely high and sparse, sometimes reaching millions which can be unmanageable in many practical applications due to computational memory and disk space. For instance, the `WikiPassageQA` dataset has 323,591 ($|\mathbf{V}|$) unique words and 244,136 ($|\mathbf{D}|$) passages. If we use the term-document representation, the dimensionality of the matrix will be $|\mathbf{V}| \times |\mathbf{D}|$ and most of the elements in the matrix will be 0 since most passages will use only a small fraction of the whole vocabulary. If term co-occurrence is used as features for the word vectors then the dimensionality of each word will be $|\mathbf{V}|$ and there will be $|\mathbf{V}|^2/2$ entries in the entire term-context matrix and it will be sparse as well for similar reasons. These sparse vectors might be computationally expensive to be used as input features to ML models— having a large number of weights to tune. Moreover, a dense vector might generalize better than these sparse vectors.

Count based word embeddings, traditionally known as **distributional semantic models** (DSM), are learned from these high dimensional representations by factorizing the word-document or word-context matrices. **Singular value decomposition** (SVD) [46] is one of the most popular approaches for this task. SVD is employed by latent semantic analysis[5] [32], COALS [117] etc. to obtain these low dimensional word embeddings and observed improved retrieval effectiveness over prior models. The main motivation for employing various techniques for reducing the dimensions of the word-context matrix is to increase the speed of computation for obtaining word embeddings (more details in Section 2.2.4). Dhillon et al. [35] in their iterative *low rank multi-view learning* (LR-MVL) used canonical correlation analysis [59] between the contexts of a given word for producing word embeddings. Lebret and Collobert [70] performed Hellinger PCA [6] transformation on the word-context matrix for the same. These models were not only faster (around 1.5 times) than previous count-based models they also gained significant effectiveness improvement over LSA, COAL in multiple NLP tasks like named entity recognition (NER) and sentiment classification.

The final count-based model that we will discuss is the widely popular `GloVe` (Global Vectors for Word Representation) by Pennington et al. [105]. The model is based on the intuition that the *ratio* of the co-occurrence of two words (rather than the co-occurrence probabilities themselves) carries the semantic information about a pair of words. They store this information in a word-context co-occurrence matrix and learns a low-dimensional representation of the words using a log-linear model. The model aims to minimize the difference of a pair of words in the latent space to the logarithm of their actual co-occurrence ratio count. The authors report improved effectiveness over other count-based models and prediction-based models in tasks like NER, word analogy and similarity.

2.2.3 Neural Network Language Modeling

Language models (LM), are the probability distribution of words over a vocabulary. The LM described in Section 2.1.2 was designed from a retrieval perspective since it calculates the likelihood of obtaining a query term from a document. LM can also be

[5]Originally built for IR, but the word embeddings can be used for NLP tasks as well

[6]This algorithm tried to minimize the distance between principal components and actual data by employing Hellinger Distance which is the Euclidean Distance analog of two probability distributions.

created by estimating the likelihood of a *target* word given its immediately preceding words (context) and these models find their application in a number of NLP and IR tasks. Ideally, we would have a complete probabilistic model with the likelihood of every word for all possible preceding words in the vocabulary. The LMs where we calculate the probability distribution of a sequence of T words by the product of the conditional probability of every word given their preceding context words can be formalized as follows:

$$P(w_1^T) = \prod_{t=1}^{T} P(w_t|w_1^{t-1}) \tag{2.6}$$

where w_t is t-th word in the sequence, w_1^{t-1} is the sequence of all preceding word to w_t starting from the first word in the window T. It has been empirically shown that a small context size of 3 (3 preceding words) is enough to give a satisfactory performance [48]. That is, words in a sequence are more dependent on context that is closer to them. We can generalize this approximation as n-gram LM (where n is the number of words in the context). Hence, Eq. 2.6 can be re-written as

$$P(w_1^T) = \prod_{t=1}^{T} P(w_t|w_{t-n+1}^{t-1}). \tag{2.7}$$

One of the fundamental problems of these LMs is the *curse of dimensionality* involved in calculating the discrete joint distributions of Eq. 2.6. For example, modeling the joint distribution of 10 consecutive words ($T = 10$) from a vocabulary of 17,000 words potentially leads to $17000^{10} - 1$ parameters that need to be trained. In an attempt to overcome this, Bengio et al. [7] were one of the earliest proponents of what is traditionally known as NNLM—they represented each word in the vocabulary using a feature vector whose size (30, 60 or 100) is much smaller than their vocabulary (17000) and expressed the joint probability function in terms of this feature vector. This drastically reduced the number of trainable parameters. Their model consists of a one-hidden layer feed-forward NN and maximizes the log-likelihood of the prototypical LM objective of Eq. 2.7:

$$J_\theta = \frac{1}{T} \sum_t \log P(w_t|w_{t-n+1}^{t-1}) \tag{2.8}$$

where $P(w_t|w_{t-n+1}^{t-1})$, i.e. the probability of the current word given its n preceding words, is calculated by the softmax layer:

$$P(w_t|w_{t-n+1}^{t-1}) = \frac{e^{z_t}}{\sum_{j=1}^{|\mathbf{V}|} e^{z_j}} \tag{2.9}$$

where z is the intermediate representation of the words. J_θ is also known as cross-entropy loss. Their model consists of three main components as can be seen from Fig. 2.2 which forms the foundation of most of the current NNLMs and word embedding models.

- **Word Embedding Layer**: The first layer is generally a word embedding matrix (or word feature matrix) using the indices of the context word. The word embedding matrix is trained by backpropagation. During inference, the embeddings are looked up from this matrix.

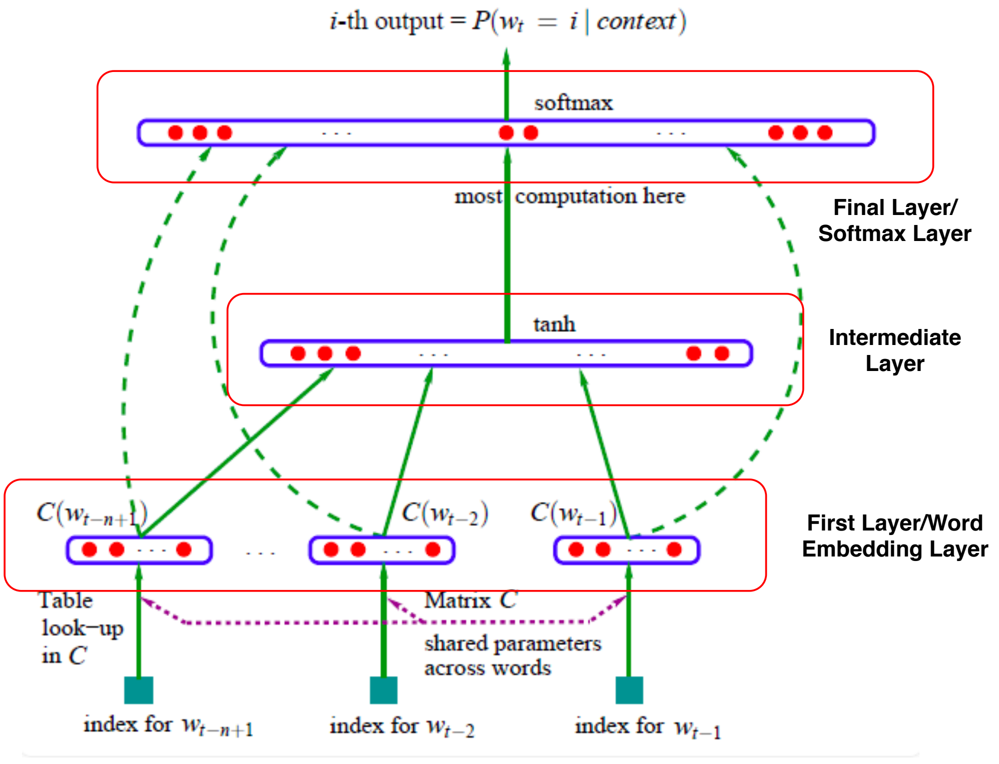


Figure 2.2: A pictorial overview of the NNLM adopted from Bengio et al. [7]. The first layer is the word embedding layer that is usually a matrix whose weights are learned by back-propagation. The matrix, after training, contains the embeddings of all words in the vocabulary which can be looked up during inference. The intermediate layer is a fully-connected layer with non-linearity. And the final layer is a softmax layer that provides a probability distribution over the words. The objective is to have the highest probability for the target word.

- **Intermediate Layer(s)**: One or more layers that produce an intermediate representation of the input. For example, a fully connected layer with a non linearity as used by Bengio et al. [7], convolutional neural networks (CNN) [107], recurrent neural networks (RNN) as used by Mikolov et al. [84], long short term memory (LSTM) RNNs as used by Sundermeyer et al. [135], Kim et al. [64] etc.

- **Softmax Layer**: The final layer which provides a probability distribution of the words in a vocabulary given that context. That is to say, the model predicts the target word given its context words. This forms *the underlying idea for prediction-based word embedding models*.

Bengio's NNLM outperformed state of the art non-neural models in run-time (100 fold speed up) and perplexity[7] in the language modeling task. However, the last softmax layer can still be computationally very expensive for large vocabularies. Most of

[7]Perplexity is a measure of how well a probability distribution predicts a word. A low perplexity indicates good performance.

the efforts during 2003 to 2008 revolved around trying to speed up training time of NNLMs either by various techniques like employing hierarchical softmax[8]in the final layer [95] or using a log-bilinear model [93] (which is faster than the log-linear model of Bengio et al. [7]'s NNLM) [9] or a combination of both [94]. These works focused on the task of language modeling—they just used the words in the left context to predict the target word—and considered word embeddings (distributed representation of words) as an interesting by-product.

2.2.4 Prediction Based Word Embeddings

Collobert and Weston [27] shifted the focus towards the specific purpose of learning word embeddings by training them on multiple downstream tasks like NER, POS tagging, semantic role labeling (SRL). The final layer produced a probability distribution over the classes of the task instead of words in the vocabulary. Since labeled training data is expensive, they also proposed a model that can learn word embeddings from the unlabeled text. This was similar to prior NNLMs like [7]. However, they introduce two noteworthy improvements:

- They used both *preceding* (left) and *after* (right) contexts to predict the target word compared to just using the left context[10]. The objective of the prediction based word embedding models is to predict the target words given the context words from both sides.

- They also employed a faster training objective than the cross-entropy based loss function of Bengio et al. [7] which tries to maximize the probability of the target word given the left context. They create incorrect word sequences by replacing the target word with an incorrect word, randomly sampled from the vocabulary V. Their model is trained to output a higher score $f_\theta(x)$ for the correct sequence than that $f_\theta(x^-)$ for the incorrect one. This technique, popularly known as **negative sampling**, has been used by a number of works [87, 13] and has shown to speed up training by avoiding costly operations like calculating cross-entropies and softmax. Their objective function can be written as:

$$J_\theta = \sum_{x \in S} \sum_{x^- \in \mathbf{V}} \max(0, 1 + f_\theta(x) - f_\theta(x^-)) \tag{2.10}$$

where S is the set of word sequences. This was the first time LM were trained on large scale training data of vocabulary size 130000 (compared to 17000 of Bengio et al. [7]) and the resulting word embeddings carried syntactic and semantic meaning while simultaneously improving accuracy on the downstream NLP tasks of NLP, POS tagging and SRL.

In 2013, Mikolov et al. [88] analyzed the word embeddings obtained as a by product of training an recurrent neural network-LM (RNN-LM) [84][11] and surprisingly

[8]If we arrange the words in a hierarchical binary tree structure, we can calculate the probability of each word by considering the path leading up to that word in the tree. This can exponentially speed up calculations.

[9]These are special cases of *log-linear* models.

[10]Prior works focused on the task of language modeling and hence they can only consider left context to predict the target word. Models whose main goal is to create word embeddings do not have any such restrictions.

[11]RNN-LM is an NNLM that emloys an RNN as its intermediate layer.

observed semantic relationship in the embeddings which could be captured using simple algebraic operations. For example, *vector*(King) - *vector*(Man) + *vector*(Woman) lead to a vector which was very close to *vector*(Queen) in the embedding space[12]. Furthermore, most prior LMs and word embedding models were computationally expensive and were not successfully trained on $|\mathbf{V}|$ in the order of millions. Hence, with the goal of scaling up these techniques (so that the new model can be trained on billions of words in lesser time) while preserving the above mentioned linear relationships in the emebeddings, Mikolov et al. [85] introduced two new models for learning embeddings, which came to be known as `word2vec` [13]. The two models were called the **continuous bag-of-words** (CBOW) and **skip-gram** (SG) models. These log-linear models differed in their training objective - CBOW uses left and right context words to predict the target words while SG uses the target words to predict the left and right context words. The training objective can be summarized in Eq. 2.11 and Eq. 2.12 respectively:

$$J_{CBOW} = \frac{1}{T}\sum_{t=1}^{T}\log P(w_t|w_{t-n},...,w_{t-1},w_{t+1},...,w_{t+n}) \tag{2.11}$$

$$J_{SG} = \frac{1}{T}\sum_{t=1}^{T}\sum_{-n\leq j\leq n, n\neq 0}\log P(w_{t+j}|w_t). \tag{2.12}$$

As can be seen, the CBOW objective function is similar to that of the LM objective (Eq. 2.8) except that it takes the right context into account as well. The SG objective sums the log probabilities of the surrounding words given the target word. These models were two times faster than existing LM and word embedding models since they forgo the costly intermediate layer of those models (more details on the architecture in Chapter 3). The word embeddings were significantly more accurate than prior models in a number of word relationships tasks. They followed up their work with a variant [87] of the models that use negative sampling (SGNS) instead of hierarchical softmax for the final layer and objective function which achieved a speedup of about 10%. The major contribution of Mikolov et al. [85, 87] is creating computationally efficient models that can be trained on data with an unprecedented vocabulary size of 692000. This resulted in embeddings with greater quality and carrying more information.

Among the more recent contributions to prediction-based models for word embeddings we mention the work of Cao and Lu [13] - their model (more details in Chapter 3) leverages sub-word information of words and uses a convolutional layer to extract higher-level semantic information of words in context, both of which are used to predict the target word. The idea of using CNN for the language modeling task (and consequently word embeddings) has been explored before ([27, 107, 31]) where the recent models outperformed RNN-LMs in the LM task. The accuracy of these word embeddings learned using CNN, however, do not outperform those obtained using

[12]Recently Nissim et al. [99] has shown that these analogy relationships hold because of certain constraints in the implementation of [88] which does not let the final vector be equal to *vector*(King). This might lead to some apparent biases in the embedding space that is not actually present in the dataset.

[13]After they released their codes for the models in a repository with the same name - `https://code.google.com/archive/p/word2vec/`.

LSTM-LM[14]. Nonetheless, CNNs have the advantage of having simpler architecture than RNNs or LSMTs with lesser parameters and can also be trained in parallel to speed up computation. Bojanowski et al. [9] in their **FastText**[15] model, used sub-word information for learning *n-gram* embeddings (which can be concatenated to form word embeddings) and noted better accuracy than SGNS in German, French and Spanish. Their intuition was that words have some information encoded in the morphological structures as well. Cao and Lu [13] combined the above two ideas—using sub-word information and a convolutional layer to learn context representation—and applied to the task of learning word embeddings. Their IWE embeddings not only performed better in the NLP tasks of word analogy and word similarity over `word2vec` (SGNS, CBOW), `GloVe` and FastText, the embeddings were also more robust to the size of training data—their effectiveness did not decrease considerably when the vocabulary size was decreased.

2.2.5 Comparison among Count-based and Prediction-based Word Embedding Models

The popularity of both `GloVe`[16] and `word2vec` in IR and NLP research [75, 11] can be attributed to the fast training of the models, the small size of the embeddings and the publicly available codes. Moreover, various labs and research groups have trained a number of word embedding models (`word2vec`, `GloVe`, FastText) on large corpus like Wikipedia dump, Gigaword[17], Common Crawl [18] etc. to obtain word embeddings for millions of words and released them online[19]. These publicly available embeddings are commonly known as *pre-trained* word embeddings and are widely used in IR and NLP tasks. Two questions arise from the background information on the word embedding models:

Are the count-based and prediction-based models inherently similar? Although apparently, count and predict models use different algorithms to learn word representations—count and predict—both types of model fundamentally act on the same underlying statistics of the data, i.e. the co-occurrence counts between words. Furthermore, Levy et al. [74] considers `GloVe` to be a prediction based model since it is trying to predict the representation of the target word even though it is clearly factorizing a word context co-occurrence matrix. Levy and Goldberg [73] demonstrate that `word2vec` implicitly factorizes a word-context PMI matrix. Hence, it cannot be said that count-based models and prediction based models are fundamentally different and indeed share some inherent similarities.

Is there one-word embedding model that is better than the rest for all tasks? Although `GloVe` and `word2vec` are shown to perform better than most other models,

[14]LSTM-LM is an NNLM with an LSTM as the intermediate layer.

[15]See https://research.fb.com/fasttext/.

[16]See http://nlp.stanford.edu/projects/glove/.

[17]See https://catalog.ldc.upenn.edu/LDC2011T07.

[18]See http://commoncrawl.org/the-data/get-started/.

[19]See https://github.com/chakki-works/chakin for a comprehensive list of widely popular pre-trained word embeddings

we cannot say that there is one model that produces better embeddings than the rest. Prediction based word embeddings have also shown to substantially outperform traditional count-based models. There has been some research that has compared count-based models and prediction based models. Baroni et al. [5] showed that, in nearly all tasks, predict models consistently outperform count models, and therefore provided us with a comprehensive verification for their supposed superiority (however, this was before the introduction of `GloVe`). Pennington et al. [105] showed that `GloVe` outperforms SGNS in tasks such as word analogy and Named Entity Recognition. Levy et al. [74] compared SVD, SGNS, and `GloVe` by training the models on a dump of the English Wikipedia and evaluating the obtained embeddings in the task of word similarity and word analogy. They showed that SVD performed best on similarity tasks, whereas SGNS performed the best in analogy tasks. They also mention that tuning the hyper-parameters of the models is often more important than choosing between models since with properly tuned hyper-parameters no model has a significant advantage over the others. SGNS outperforms `GloVe` in all tasks of Levy et al. [74]. Hence, the general conclusion is that a single model is not better in all tasks, rather the performance is dependent on the task, training data and the proper tuning of hyper-parameters.

In the following subsection, we will provide a brief overview of how word embeddings are typically employed in IR research, specifically in document ranking.

2.2.6 Word Embeddings for Document Ranking

Document ranking in ad-hoc retrieval is essentially the task of finding the relevance scores of documents with respect to a query and then ranking the documents based on the score. To achieve the same we need to generate a representation of the query, a representation of the considered document and calculate the similarity scores between the representation to estimate the relevance of the document to the query. In traditional IR models, the queries and documents are represented in the form of bag-of-words. The more recent neu-IR models *learns to rank* the documents with respect to a query by forming their own representations.

These word embeddings of query and document terms form the input to the neu-IR models where they are subsequently passed through neural network architectures to form query and document representations. These representations are finally used to calculate the relevance score for the query-document pair by a ranking function:

$$S(\mathbf{d},\mathbf{q}) = g(\psi(\mathbf{q}),\phi(\mathbf{d}),\eta(\mathbf{d},\mathbf{q})). \tag{2.13}$$

The above equation has been adopted from [51, 91, 148] where $\mathbf{d}$ and $\mathbf{q}$ are the word embedding input representation of the document and query, ψ and ϕ are functions that form higher lever representations of $\mathbf{d}$ and $\mathbf{q}$, η is the interaction function that extracts feature from $\mathbf{d}$ and $\mathbf{q}$ together and g is finally the scoring function which computes the relevance score of the $\mathbf{d}$-$\mathbf{q}$ pair. Fig. 2.3 shows a graphical representation of the general architecture of these neural IR models. The learning objective of these neural models is to minimize the difference between relevance labels of the query-document pair and the predicted ranking score obtained using S. In other words, $S(\mathbf{d},\mathbf{q})$ should be higher for a document that is more relevant to a query than the one which is less or non-relevant to the same query. This can be expressed in a loss function that is used to train these neural models. The input or the word embedding layer may or may not be considered a part of the *trainable* components of the neural network architecture. Typically they are not re-trained and just used as a basic input layer so that different pre-trained word embeddings can be employed in the same model.

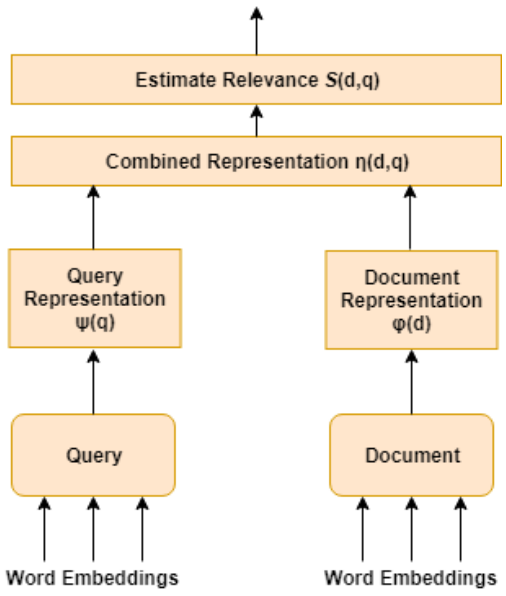

Estimate Relevance S(d,q)
Combined Representation η(d,q)
Query Representation ψ(q)
Document Representation φ(d)
Query
Document
Word Embeddings
Word Embeddings

Figure 2.3: A pictorial overview of the neural IR model inspired from [90]. This is a hybrid approach and in general neural IR models might have either or both the representation and interaction based components.

Neural IR models are often categorized [51] into three types based on the manner they model the input representations of queries and documents - **representation-based**, **interaction-based** and **hybrid** approaches. Representation-based models [60, 127, 126, 103] employ neural network architectures, ψ and ϕ, to create good representations of the query and documents separately from their word embeddings which are ultimately combined by similarity function g in the last layer. Interaction-based models [50, 104, 39, 137] defines an interaction matrix η to capture local interactions between $\mathbf{d}$ and $\mathbf{q}$ like exact matching of n-grams or cosine similarity of queries and sentences of the documents etc. The output of η is then passed through a neural network to be similarly used by g to find the relevance score. Hybrid models use both representation-based (ψ and ϕ) and interaction-based (η) components separately [91] or one-after-another [145] to finally obtain the relevance score.

To summarize, word embeddings are used as the input layer for these neu-IR models to retrieve a ranked list of documents or passages. The semantic information carried by word embeddings is also used to find QE terms to a given query. In the following sections, we first provide a brief introduction to QE approaches followed by a discussion on how word embeddings are typically used for the same.

2.3 Query Expansion

The main idea of QE technique is to let the system help (automatically or semi-automatically) by adding terms to the original query to close the semantic gap between a user's query and her information need. The performance of the QE approach can only be judged by comparing the retrieval effectiveness of the IR model with and without the QE terms generated by the approach. We can say QE is effective when it improves performance over the no-expansion IR model. QE approaches can be thesaurus based where expansion terms are generated from a domain-specific thesaurus or generic thesaurus like WordNet (where QE is done by adding semantically similar terms called synsets). Sometimes external text collections like Wikipedia dumps are also used.

Another QE approach involves a first round of retrieval to retrieve documents in response to the original query. Following this round, a feedback is provided on which documents are relevant among the retrieved ones. The idea is that words that occur more frequently in relevant than non-relevant documents are more important and thus used as QE terms to the original query. This approach is also known as **relevance feedback** (RF) - an idea that was first introduced by Rocchio [116] as a VSM. Their goal was to obtain an optimal query vector from the original query vector using the RF from users. The general overview of their approach can be summarized in the following steps:

1. User formulates a query.
2. The IR model returns a ranked list of documents - *the first round of retrieval*.
3. The user provides feedback on which documents are relevant and non-relevant to her query.
4. The system forms a better representation of the query by adding important terms from the relevant set of documents.
5. The IR model returns a revised ranked list of documents.

Steps 3-5 can be repeated more than once until the ideal query representation is obtained. Most QE techniques use this general approach to find expansion terms. The relevance feedback in step 3 can be of three types — *explicit* [119, 54] - where the user explicitly mentions the relevance of retrieved documents; *implicit* [20, 163] - where the user activities on retrieved documents like clicks or their social media information are used to implicitly interpret which documents are relevant; and lastly, the process of relevance feedback can be automated by considering a number of top-ranked documents in step 2 to be relevant, a process which is also called pseudo relevance feedback (PRF).

Croft and Harper [28] proposed **relevance model** (RM) which gives us the expected distribution of words in the relevant documents. In the LM of Eq. 2.6, queries are fixed samples and documents are ranked based on their probability of generating the sample. The notion of PRF can be incorporated in the LM if we consider the query terms to come from the probability distribution of the RM, instead of fixed samples. The relevant documents are larger samples from the same model. Given the RM $\mathcal{R}$, documents can be ranked according to the KL divergence [160][20] score of document LM ($\mathcal{D}$) which is the probability distribution of words in individual documents) and $\mathcal{R}$ as follows:

$$score(\mathbf{d},\mathbf{q}) = -KL(\mathcal{R}||\mathcal{D}) = -\sum_{w\in\mathcal{V}} P(w|\mathcal{R}) \log \frac{P(w|\mathcal{R})}{P(w|\mathcal{D})} \tag{2.14}$$

The main intuition is that we are trying to find documents with similar distribution to the RM - the expected distribution of relevant documents. The question that arises is how to estimate the RM without any relevance information. We can estimate $\mathcal{R}$ as the probability distribution over the vocabulary based on the query terms which, as discussed, are a small sample from $\mathcal{R}$. This is the conditional probability of each word given the query terms and can written as follows:

$$P(w|\mathcal{R}) \approx p(w,\mathbf{q}_1,\mathbf{q}_2,...,\mathbf{q}_n)/p(\mathbf{q}_1,\mathbf{q}_2,...,\mathbf{q}_n) \tag{2.15}$$

The above equation gives high probability to words that are expected in the same documents as query terms. Assuming exchangeability of words (that is the order in which the words are observed does not matter), we can use de Finetti's theorem [21] to compute the above joint probability distribution as

$$p(w,\mathbf{q}_1,\mathbf{q}_2,...,\mathbf{q}_n) = \sum_{\mathcal{D}\in\mathcal{C}} P(\mathcal{D})P(w|\mathcal{D}) \prod_{i=1}^{n} P(\mathbf{q}_i|\mathcal{D}) \tag{2.16}$$

where $\mathcal{C}$ is a collection of document LMs, $P(\mathcal{D}) = 1/|\mathcal{D}|$ is the uniform prior over documents and $\prod_{i=1}^{n} P(\mathbf{q}_i|\mathcal{D})$ is the QL score of a document. Hence, RM incorporates PRF in LM in the following manner:

1. User formulates a query.
2. The documents are ranked according to their QL score to obtain the weights needed to compute the RM.
3. Top $fbDocs$ documents are selected to be in the collection set $\mathcal{C}$. This is the PRF assumption of relevance.
4. Calculate RM probabilities $P(w|\mathcal{R})$ with the estimates of Eq. 2.15 and Eq. 2.16 and select top $fbTerms$ expansion terms based on the probabilities and add them to the original query.
5. Re-rank the documents using the KL divergence score of Eq. 2.14.

Comparing the above steps with that of the general RF approach we can clearly see how the PRF based approach varies in steps 2-5. There are different variations of RMs employed in literature out of which we use the popular RM3 version which is not only one of the most effective approaches of QE [25] but is also used a strong baseline IR

[20]Kullack - Leibler (KL) divergence measures the difference between two probability distributions. Higher the KL divergence score, more apart the two distributions are

[21]In probability theory, de Finetti's theorem states that exchangeable observations are conditionally independent relative to some latent variable.

model. RM3 regularizes the RM $P(w|\mathcal{R})$ by interpolating it with the term frequencies of the original query as follows:

$$P(w|\mathcal{R}') = fbOrigWeight\frac{tf(\mathbf{q}_i, \mathbf{q})}{|\mathbf{q}|} + (1 - fbOrigWeight)P(w|\mathcal{R}) \tag{2.17}$$

where $fbOrigWeight$ is the interpolation weight. The above equation also represents the expanded query LM and can also be seen as a query reformulation technique. The first part of Eq. 2.17 assigns a weight to each query term based on its frequency in the query $\mathbf{q}$ (query LM) and the second part is the weight of top $fbTerms$ expansion terms (expansion LM) calculated using the above algorithm. $fbOrigWeight$ is used to weigh the comparative contribution of the original query terms and the expansion terms. For our thesis, we use the RM3 implementation of `Indri` as a baseline. RM3 model suffers from one drawback—it involves 5 hyper-parameters ($fbTerms$, $fbDocs$, $fbOrigWeiht$, $fbMu$ for the first round and μ for the second round of retrieval) that need to be optimized for testing compared to 1 (μ) of QL with Dirichlet smoothing. This makes tuning of RM3 hyper-parameters more tedious. Fig. 2.4 shows a graphical representation of the QE pipeline using RF/PRF.

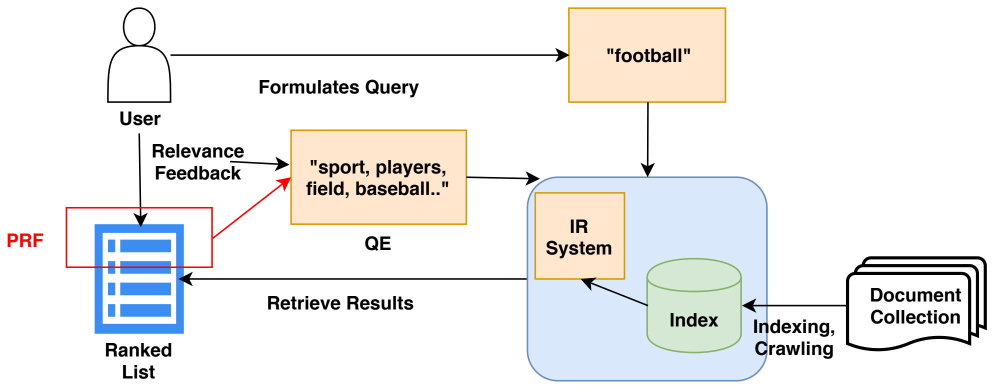


Figure 2.4: A pictorial representation of a QE using relevance feedback from users or using PRF where the top documents in the ranked list are assumed to be relevant

In the recent trends of the QE approach, with the popularity of word embedding models, expansion terms are normally generated as the words in the vocabulary with the highest cosine similarity to the query embeddings [68, 159, 118, 101, 37]. The weights of the expansion terms are typically their cosine similarity score with the query terms. More details on QE techniques using word embeddings are discussed in section 2.4.

2.4 Query Expansion using Word Embeddings

Word Embeddings, as discussed in Section 2.2, are a low dimensional numerical representation of words in a vocabulary. The latent space in which the words are embedded is used to capture the semantic and syntactic properties of these words. Words having similar meanings are normally placed in similar regions of the latent space. Hence, various similarity or distance measures can be used to find words that are nearest neighbors or similar to a particular word. The most commonly used similarity measure is the cosine similarity (Eq. 2.5).

Since QE is essentially finding semantically similar words to a query, word embeddings can be leveraged to find the expansion terms. That is to say, we can find the nearest neighbors of the query terms in the embedding space using cosine similarity and weigh them according to their cosine similarity score. This can be seen as the expansion LM of the second part of Eq. 2.17. Following which we can interpolate them with the original query LM and use the final expanded query LM to retrieve documents or passages with IR models like QL or BM25. This is the basic framework employed by a number of words [68, 159, 118, 101, 37] that uses word embeddings to find QE terms. The word embedding models for this task can either be trained on the entire corpus - **global word embedding models** or can be trained on a query-specific subset of a corpus **local word embedding models**. In this section, we discuss the two approaches.

2.4.1 Global Word Embedding Models

To our knowledge, Roy et al. [118] was the first to use word embeddings to generate expansion terms for the ad-hoc retrieval task using the above framework. Prior to that word embeddings of query and documents were used as input to neural network architectures to find latent semantic representations of the query and documents [60, 127, 126, 50] (Section 2.2.6). The goal of Roy et al. [118] was to observe if the K-nearest neighbor terms of each query terms in the embedding space were good expansion terms i.e. whether they improved retrieval effectiveness. The pooled, $K \times |\mathbf{q}|$ (K times the number of query terms) candidate expansion terms were ranked according to their mean cosine similarity score with all query terms, out of which k were chosen as expansion terms. They trained CBOW model on the entire corpus (the TREC disk 4 and 5 collections and the web WT10G collection) to obtain the embeddings. They observed that although the nearest neighbor expansion model outperforms the *no expansion* model, they were significantly inferior to the popular RM3 baseline. However, one advantage of this model is that only one round of retrieval round is necessary for the entire process.

Kuzi et al. [68], Zamani and Croft [159] extended the above global word embedding based query expansion approach with RM1 and RM3 respectively—their models were a linear combination of QE using word embeddings and QE using PRF. They observed significantly improved retrieval effectiveness over the PRF based baselines of RM1 and RM3. PRF approaches generate expansion terms from documents relevant to the original query, a piece of information that is not utilized if we consider the approach of Roy et al. [118]. These works show that the knowledge of relevance is important while generating the expansion terms - the main underlying intuition behind the local word embedding models. A graphical overview of QE using the global word embeddings approach is presented in Fig. 2.5.

2.4.2 Local Word Embedding Models

Diaz et al. [37] followed a similar k-nearest neighbor approach with one important difference. They pointed out that word embeddings trained using the global corpus

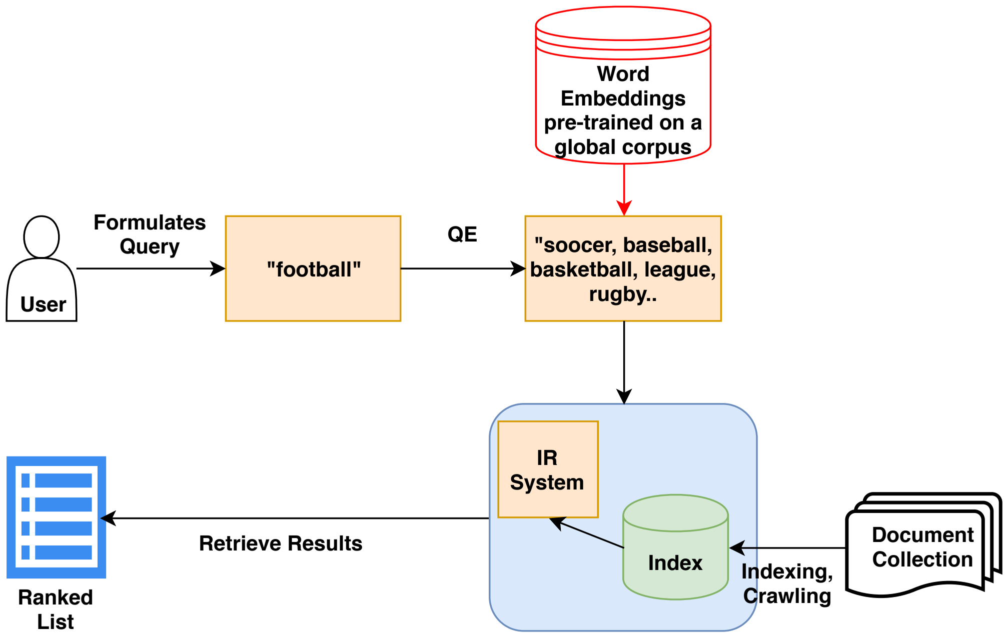


Figure 2.5: A pictorial overview of the retrieval pipeline with QE using global word embeddings which is also employed in our experiments

statistics will be different from those trained on a query-specific subset of the same corpora. They proposed to train the word embeddings on the query-specific documents instead of the whole collection. In order to obtain the topic-specific documents, they perform an initial round of retrieval using the original query out of which the top 1000 documents were considered query-specific. Finally, they train CBOW on the 1000 documents retrieved for each of the queries. The main intuition is that the documents retrieved by the original query give a notion of relevance for the query (similar to PRF based approaches) in addition to the fact that their unigram distribution is different from that of the entire corpus. This initial round of retrieval can be performed either on the target corpus (which contains the relevant documents or answer passages) or on a larger external corpus. As the authors mention in their work, the embeddings from the external corpus have the advantage of containing more information from a larger variety of topical material than what the embeddings from the target corpus contain [36]. For them, only QE using the embeddings trained on the external corpus significantly outperformed the global word embedding model - QE terms obtained using word embeddings pre-trained on an unconstrained corpus. An overview of QE using word embeddings from a topic-specific corpus is shown in Fig. 2.6.

The idea of using an external corpus was implemented much earlier by Xu et al. [150] where the authors leveraged text from Wikipedia to perform the first round of retrieval of RM and observed improved performance over the baseline RM. The local embedding approach of Diaz et al. [37], however, suffers from an important drawback. The word embedding model has to be trained in an *online* manner on the top 1000 retrieved documents for every query. That is, not only two rounds of retrieval is required, the word embedding model also has to be trained after the first round of retrieval for each query. This makes the approach expensive and inefficient time-wise. A solution

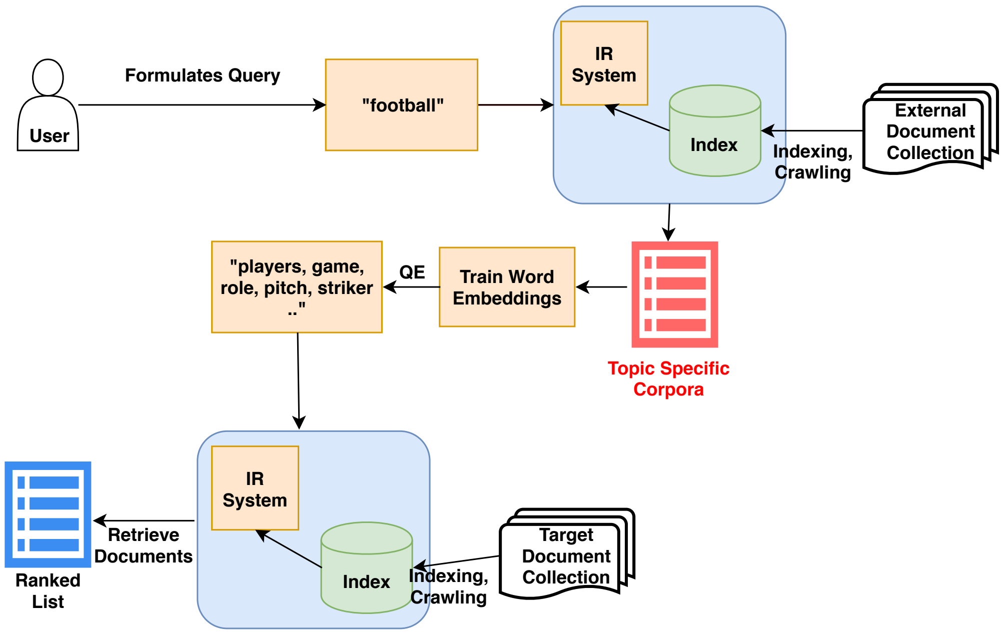


Figure 2.6: A pictorial overview of the retrieval pipeline with QE using word embeddings trained on a topic-specific corpus. In this image, the query-specific corpus is obtained from an external collection. It can be also obtained from the target document collection, where the first round of retrieval will retrieve documents from the target corpus.

to this, as mentioned by the authors, is to try to do the word embedding training in an *offline* manner before the retrieval round. Another solution can be to retrieve less number of documents in the first round of retrieval to speed up the process of training the word embedding models. Table 2.1 provides a comparative summary of the mentioned query expansion models.

As we have discussed in Chapter 1, the task of passage retrieval involves retrieving a limited number of candidate passages to each query, which forms the query-specific data for training the local word embedding models. However, the vocabulary size in the candidate set of passages is considerably small which can be problematic for training the models as the resulting embeddings might fail to capture semantic information. A similar issue was faced by Ould-Amer et al. [101] in their local expansion model for the task of CLEF Social Book Search[22] [66]. They trained the word embedding models on one short document for each query to generate the expansion terms. The model could not outperform the no expansion baseline since the `word2vec` model did not have sufficient data to train, leading to embeddings of poor quality and in turn, it produces expansion terms that cannot increase retrieval efficiency. Hence, the research in our thesis has two objectives:

- To observe if expansion terms produced by local embedding models (target and external) improves in retrieval performance over those produced by the global

[22]See http://social-book-search.humanities.uva.nl/#/suggestion

Table 2.1: Comparison of various query expansion models. We identify the models by local (L) or global (G) depending on whether they use an entire corpus or query-specific subset of a corpus (with relevance assumption) for generating the expansion terms. Off indicates word embedding training is done before any retrieval round and On indicates the training is done during the retrieval.

QE model	**Corpus**	**# Retrieval rounds**	**Relevance Assumption**	**#WE training**
RM3	L	2	Yes	-
Roy et al. [118]	G	1	No	SGNS, Off
Kuzi et al. [68]	G + L	2	Yes	CBOW, Off
Zamani and Croft [159]	G + L	2	Yes	`GloVe`, Off
Diaz et al. [37]	G	1	No	`GloVe`, Off
Diaz et al. [37]	L (target)	2	Yes	CBOW, On
Diaz et al. [37]	L (external)	2	Yes	CBOW, On

embedding model in the task of answer passage retrieval to open-domain QA (**RQ1**).

- To focus on the retrieval efficiency of the QE terms produced by CBOW and IWE models when trained using a limited vocabulary. Consequently, our goal is also to understand the function of various components of the word embedding models in such a data-adverse condition (**RQ2**).

Chapter 3

Methodology

In chapter 1, we have discussed our arguments for employing the retrieval pipeline introduced by Diaz et al. [37] in this new context of answer passage retrieval. In this chapter, we discuss the methodology in detail. Specifically, we discuss the architecture and choice of our word embedding models CBOW and IWE (Section 3.1). We follow this up by a discussion on the difference in training methodology of the models when they are trained on entire corpus (global) and query-specific (local) corpus (Section 3.2). Next, we discuss, the methodology of QE using the trained word embeddings (Section 3.3) followed by an introduction of our dataset employed for our task of answer passage retrieval using QE (Section 3.4). Finally, we revisit our research questions in detail (Section 3.5). An overview of our methodology and this experiment is presented in Fig 3.1.

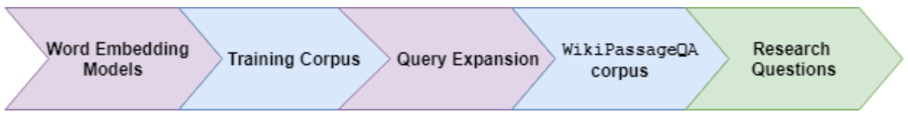


Figure 3.1: An overview of the methodology adopted in our thesis.

3.1 Word Embedding Models

Here we describe the model architecture of our two word embedding models. The section builds on the discussion of Section 2.2.

3.1.1 CBOW

CBOW or the continuous bag-of-words word embedding model is one of the variants of the `word2vec` algorithm introduced by Mikolov et al. [85] (the other being the skip-gram model). In order to understand the CBOW model let us consider the sentence (S) — "We will play football after school." as an example. The objective of CBOW is to predict a target word ('football') from context words ('will', 'play', 'after', 'school') which are on both of its sides. The CBOW architecture is depicted in detail in Fig. 3.2.

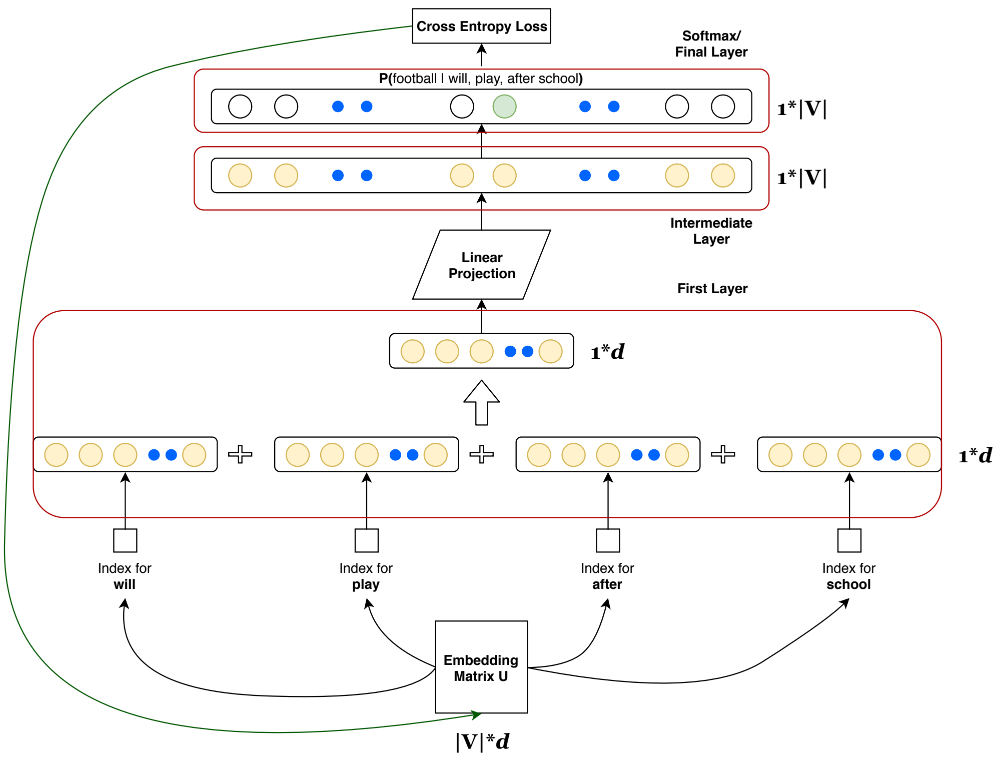


Figure 3.2: CBOW model for learning word embeddings.

Training To understand how the model is trained, let us consider the above sentence comes from a vocabulary of size $|\mathbf{V}|$ and the dimensions of the word embeddings is d. The training procedure takes the following steps:

1. All words are in the vocabulary are randomly initialized in an embedding matrix $\mathbf{U}$ whose size $|\mathbf{V}| \times d$.
2. Let at the current instance of training we have encountered sentence S and we are predicting the word 'football' which is the j-th word in the vocabulary.
3. The weights of the words in its context are looked up from $\mathbf{U}$ and added together to form the $1 \times d$ representation of the context which forms the first layer of the model.
4. This layer is then transformed through a linear projection to form the $1 \times |\mathbf{V}|$ intermediate layer.
5. The final layer produces a probability distribution over all words in the vocabulary by softmax (Eq. 2.9) computation. We are interested in the probability of the j-th word 'football'.
6. Lastly, we calculate a cross-entropy loss which is given by the equation

$$J_{CBOW} = \log P(\text{football}|\text{will, play, after, school}). \tag{3.1}$$

The generalized objective function is the one shown in Eq. 2.11 of Section 2.2.4.

7. The loss is then back-propagated throughout the model to update the weights of the linear projection layer and the word embeddings in matrix **U**.

Steps 2-7 is repeated for all sentences in the vocabulary where each word in every sentence is predicted as the target word (Fig. 3.3). Thus the loss for one iteration is the cumulative loss of predicting every word in every sentence. The training is done for a certain number of iterations until the loss is satisfactorily minimized. The trained embeddings in **U** of all words in the vocabulary are used for inference and other downstream tasks (finding QE terms in our case).

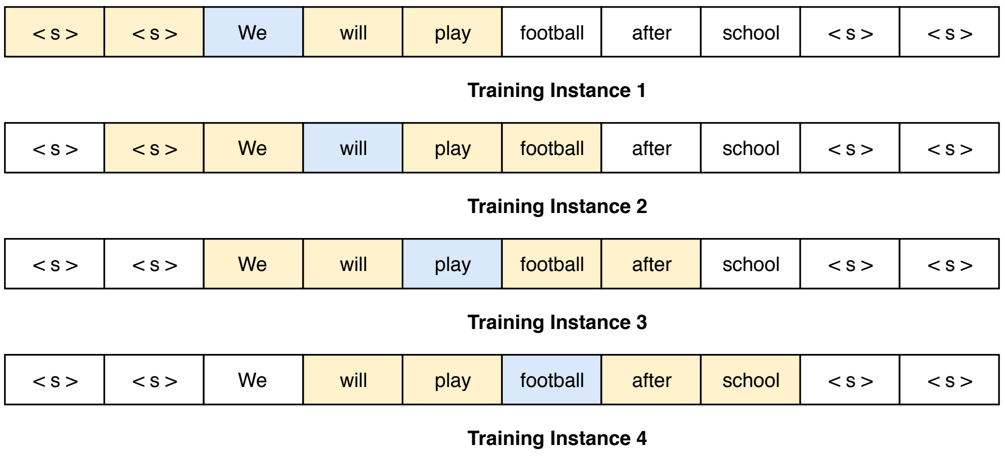

< s > < s > We will play football after school < s > < s >
Training Instance 1
< s > < s > We will play football after school < s > < s >
Training Instance 2
< s > < s > We will play football after school < s > < s >
Training Instance 3
< s > < s > We will play football after school < s > < s >
Training Instance 4

Figure 3.3: An example depicting the first four training instances of our sentence S. The blue box represents the target word and the yellow boxes represent words in the context used for predicting the target word. $< s >$ denotes a start/end of sentence token. The blue box of training instance slides over all words in the sentence. This is repeated for all sentences in the vocabulary.

Drawing parallels with the NNLM model of Bengio et al. [7] depicted in Fig. 2.2, we can see that CBOW has a similar architecture with all three layers. The difference lies in the intermediate layer which is a linear projection for CBOW in contrast to the fully connected non-linear layer of NNLM. CBOW also has the computationally expensive softmax computation in the final layer which can be fastened by introducing negative sampling. The intuition is that instead of computing softmax over the entire vocabulary we randomly sample N^- negative words from the vocabulary and calculate the loss based on these words only. The negative sampling objective for one training instance becomes

$$\begin{aligned} J_{CBOW^{NS}} = {} & \log P(w_t | w_{t-n}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+n}) \\ & + \sum_{i=1}^{N^-} \log -P(w_i | w_{t-n}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+n}). \end{aligned} \tag{3.2}$$

CBOW with negative sampling is depicted in Fig. 3.4. The rest of the architecture remains the same. We use CBOW with negative sampling for our thesis owing to its faster computation.

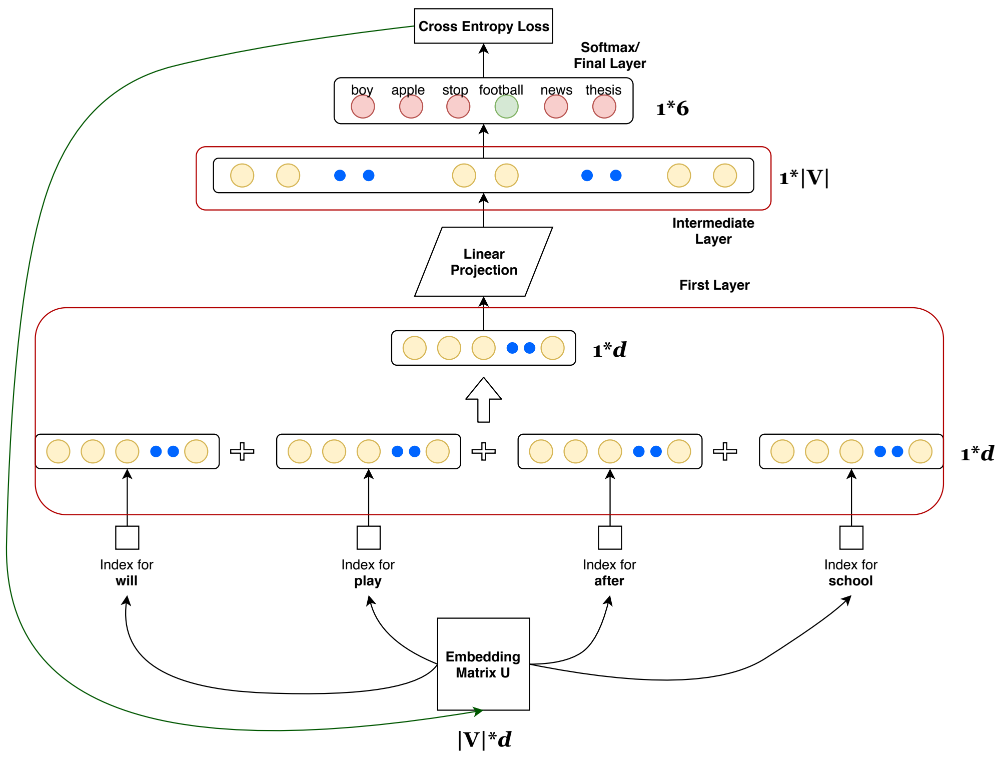


Figure 3.4: CBOW model for learning word embeddings using negative samples. In this example the number of negatively sampled words are N^{-} (‘boy’, ‘apple’,‘stop’,‘news’,‘thesis’). Instead of computing softmax over the entire vocabulary, we only need to compute softmax over the 5 negative words and our target word ‘football’.

Why CBOW A number of reasons influenced our decisions for choosing the CBOW word2vec model which we list below:

- As mentioned in Section 2.2.4, there are a number of publicly available implementations of both `word2vec` models - CBOW and skip-gram which can be easily used. Moreover, it enables the easy reproduction of the experimental pipeline.

- `word2vec` embeddings are standard representations of words for almost all NLP tasks[1] [86, 151, 79, 128] because of model simplicity and ability of the embeddings to capture semantic relationship among words. They are extensively used in the IR community as well to form query and document representations for a task like document ranking (ad-hoc search) [124, 96, 148], QEn [118, 37, 68].

- Although there is no study conclusively mentioning which `word2vec` variant is better, we specifically employ the CBOW model because Diaz et al. [37] use the same for their pipeline.

[1]There are about 10000 results for the search ‘word2vec NLP’ in https://scholar.google.nl/.

- Moreover, the skip-gram model predicts each of the context words using the target word. Thus it creates C training samples from each training instance compared to just 1 of CBOW making it a slower process [85].

3.1.2 IWE

Like CBOW, the IWE word embedding model introduced by Cao and Lu [13] is also a prediction based model with a similar task of predicting a target word given its context. IWE differs from CBOW in two aspects - the input representation and convolutional feature learning of the context as can be seen from Fig. 3.5. The input representation aims to encode useful linguistic information of the words from their sub-word representations. It consists of two pieces of information — **letter trigram features** which capture certain morphological information when similar words access overlapping feature space. For example, “player” and “playing” both share the features “#pl”, “pla”, “lay”. The letter trigrams shared between the words indicate that the words can potentially be related to each other. **Root and derivational affixes** which can be very useful for understanding the semantics and derivative morphology of a word. For example, “player”, “play”, “playing” and “player” all have the same root “play”. The various affixes provide derivational morphology that changes the meaning of the words yet they remain semantically similar because of the root word.

The convolutional layer aims to capture local contextual features from word ordering and sub-word information. CNNs are used in text classification tasks like sentiment classification [63, 45] where they are known to behave as n-gram detectors. The filters together with the max-pooling layer are known to capture important tri-grams in a sentence to make the prediction. In IR, CNNs have been used for predicting the relevance of a document with respect to a query [127, 126]. Semantically related words in documents and queries are captured by the same filters which result in a higher matching score for a document with respect to a query. In the IWE model, we use CNN in a slightly different application. In contrast to CBOW, where the target word is predicted directly from the context, CNN will learn context embedding which contains semantic information regarding the context to predict a target word. In the following section, we will discuss the IWE model in detail and elaborate on how it is trained.

Training We will consider a similar training setting like CBOW where our vocabulary is $\mathbf{V}$, the dimension of the word embeddings is d and the current training instance is sentence S where we are predicting the word ‘football’. The training methodology has the following steps:

1. Two dictionaries are built which respectively comprises of all the unique tri-grams and root affixes in the vocabulary. Let their sizes be $|\mathbf{V}_t|$ and $|\mathbf{V}_r|$. Each word is represented by the concatenation of the one-hot encoding of the tri-grams and root affixes they are made of. For example, if ‘#pl’, ‘pla’, ‘lay’ and ‘ay#’ are the 1st, 2nd, 3rd and 4th tri-grams in the tri-gram dictionary and ‘play’ is the 1st root word in the root dictionary, the word ‘play’ will be represented by the concatenation of the $1 \times |\mathbf{V}_t|$ vector $[1,1,1,1,0,0,...,0]$ and the $1 \times |\mathbf{V}_r|$ vector $[1,0,0,...,0]$.

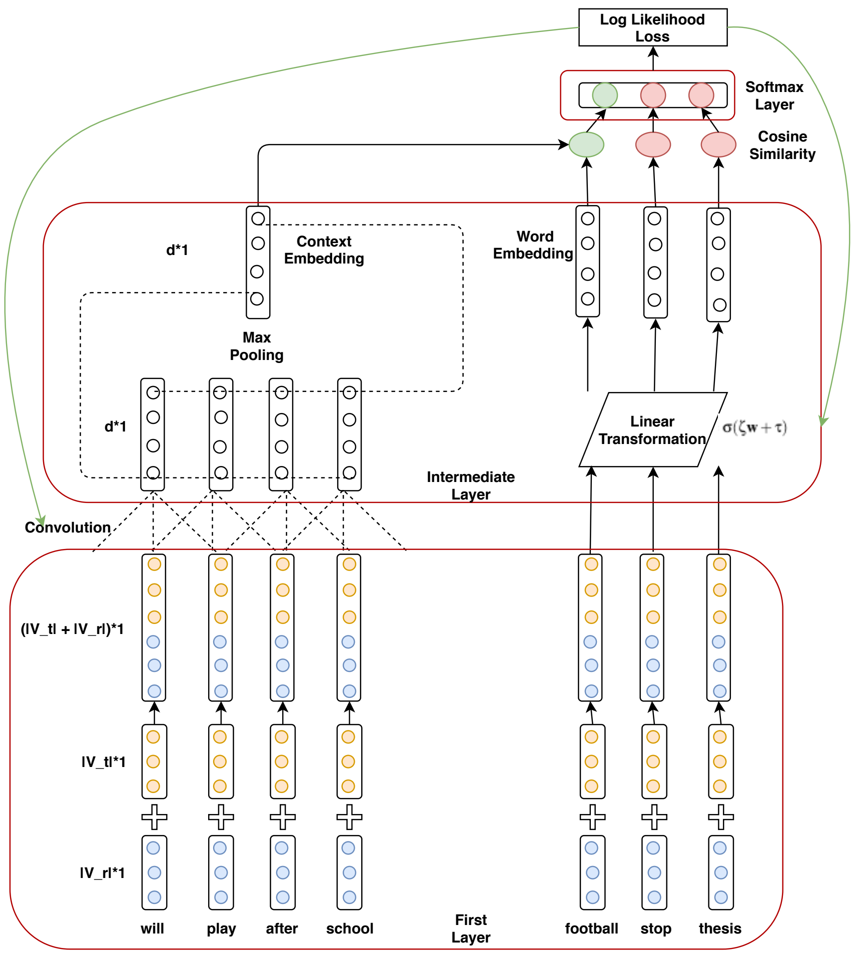


Figure 3.5: IWE word embedding model. In this example the number of negatively sampled words are $N^{-} = 2$ ('stop','thesis'). The sliding window for convolution layer $l = 3$. The objective is to maximize the similarity between context embedding and the word embedding of 'football' and minimize that with the embeddings of 'stop' and 'thesis'.

2. The target word, C words in its context and N^{-} randomly sampled negative words are represented as mentioned above.

3. For the words in the context, we thus have $(|\mathbf{V}_r| + |\mathbf{V}_t|) \times C$ matrix which is passed into the convolutional layer where we pass a sliding window of size l.

4. The number of filters f is equal to the number of output dimensions that we want, which in our case is equal to d. The size of each filter is $(|\mathbf{V}_r| + |\mathbf{V}_t|) \times l$.

5. Each filter performs the convolution operation $< [\mathbf{u}_i, ..., \mathbf{u}_{i+l-1}], f_j >$ where $i \in [0, C-l]$ and $j \in [1, d]$ on l words in the context which is given by the following:

$$y_i =< [\mathbf{u}_i, \dots, \mathbf{u}_{i+l-1}], f_j >= \sigma(W * [\mathbf{u}_i, \dots, \mathbf{u}_{i+l-1}] + \beta) \tag{3.3}$$

where $\mathbf{u}_i$ is the input representation of the i-th word, W and β are the weights of filter f_j. Thus, f_j slides over all l-words sequences in the context and produces C outputs (if we consider padding).

6. When all filters have performed the above step, we get a $C \times d$ output. This representation of convolution feature learning is able to capture local information from the sub-words and ordering of the context words.

7. To combine local information, we apply max-pooling which is given by

$$\mathbf{c}(j) = \max_{i=0,\ldots,C-1} \{y_i(j)\}; 1 \leq j \leq d. \tag{3.4}$$

Thus we get $\mathbf{c}$, which is a d dimensional representation of the context. A detailed schematic of the entire convolution process is presented in Appendix A.

8. The target word and the negative word representation are passed through a linear transformation followed by a non-linearity given by:

$$\mathbf{w} = \sigma(\zeta \mathbf{u} + \tau) \tag{3.5}$$

where $\mathbf{u}$ is the input representation of the target word, ζ and τ are the weights of the linear transformation. $\mathbf{w}$ is the embedding of the target word. Let $\mathbf{w}_j^{'}$ be the embeddings of the j-th negatively sampled word.

9. We compute the cosine similarity $s(\mathbf{w}, \mathbf{c})$ of the target word embedding $\mathbf{w}$ and the context $\mathbf{c}$. The loss function is defined as the negative log-likelihood for each pair of target word and the context representation. The loss function associated with a particular word context pair $(\mathbf{w}, \mathbf{c})$ is given by

$$l(\mathbf{w}, \mathbf{c}; \theta) = -\log p(\mathbf{w}|\mathbf{c}) \tag{3.6}$$

where $p(\mathbf{w}|c)$ is the softmax function defined over the similarity layer composed of the cosine similarity between the context $\mathbf{c}$ and each of target and negatively sampled words and θ is the parameters of the neural network. The softmax function can be defined as

$$p(\mathbf{w}|\mathbf{c}) = \frac{\exp(\gamma s(\mathbf{w}, \mathbf{c}))}{\exp(\gamma s(\mathbf{w}, c)) + \sum_j \exp(\gamma s(\mathbf{w}_j^{'}, \mathbf{c}))} \tag{3.7}$$

where $s(\mathbf{w}_j^{'}, \mathbf{c}))$ is the cosine similarity between the context $\mathbf{c}$ and the embedding of the jth negatively sampled word and γ is a temperature parameter that influences the effect of the similarity value. Thus the loss becomes

$$l(\mathbf{w}, \mathbf{c}; \theta) = \log(1 + \sum_j \exp(\gamma \Delta_j(\mathbf{w}, \mathbf{c}))) \tag{3.8}$$

where $\Delta_j(\mathbf{w}, \mathbf{c}) = s(\mathbf{w}, \mathbf{c}) - s(\mathbf{w}_j^{'}, \mathbf{c})$ is the difference of similarity between the target word and jth negatively sampled word with the context.

Then we compute the similarity between the context representation and word embeddings of the target and negative words. The objective is to maximize the difference in the similarity of the context with the target words than that with the negative terms. We do that using a negative log-likelihood loss function over the softmax distribution of the similarity scores. Similarly to CBOW, steps 2-9 are repeated for all words in all sentences in the vocabulary. Table 3.1 summarizes the difference between the two models.

Table 3.1: Comparison of models: CBOW vs IWE

	CBOW	**IWE**
#Training parameters	$2 \times \lvert\mathbf{V}\rvert \times d$	$(\lvert\mathbf{V}_r\rvert + \lvert\mathbf{V}_t\rvert) \times d +$ $l \times (\lvert\mathbf{V}_r\rvert + \lvert\mathbf{V}_t\rvert) \times d$
#Hyper-parameters	d, C, N^-, α	$d, C, N^-, l, \gamma, \alpha$
Input Dimensions	d	$(\lvert\mathbf{V}_r\rvert + \lvert\mathbf{V}_t\rvert)$
Context Embedding	Concatenation of input	Convolution over input
Output Dimensions	$1 + N^-$	$1 + N^-$
Word embedding layer	First	Intermediate

Why IWE Although IWE has a more complex model (more parameters to train and hyperparameters to tune) than CBOW, it overcomes certain short-comings of CBOW and is beneficial for our task of passage retrieval. We discuss our choice of using IWE in the following:

- IWE can be seen as an extension to the CBOW model - instead of concatenating the representations of the context words we are learning contextual features using the convolution layer. The remaining part of the model remains similar to the CBOW model wherein we try to predict a target from this representation of context.

- The input representation in the form of character tri-graph and root affixes makes it possible to have a representation of unseen query terms in the IWE model. This is because it is more than likely to have one or more constituent character tri-grams of the unseen query terms in our tri-gram vocabulary. For example, although a query term like 'particular' is not present in the training vocabulary a word like 'participate' is present. Thus, we have the tri-grams of 'participate' in our vocabulary some of which are also shared by 'particular'. Thus we can have a representation of the latter with the tri-grams it has in common with the former and pass them through the trained linear transformation layer to have their embedding. CBOW has no such provisions. Hence, if a query term is not present in the vocabulary there will be no representations for it. Moreover, the words do not come from the same root and stemming would not help either. Thus it will be omitted from the process of obtaining QE terms. We believe this property of the IWE will help in improving retrieval effectiveness using QE.

- Order of words in a sentence captures semantic information in natural language [129]. As we can see from Fig. 3.2, the representation of the context does not depend on the order in which the context words appear. Hence, the word embeddings produced by CBOW do not contain the semantic information contained in word orders. However, the context representation in IWE depends on the order as the CNN filters will produce different representations when the orders are different. And since the target word predicted from this context representation, the word embeddings will contain semantic information obtained from word orders. This will be beneficial in scenarios where the word embedding models need to be trained on limited data. When training data is less, semantically similar words might not occur in a similar context. CBOW produces similar embeddings for words when they share similar context and hence when there is less data the embeddings might not be able to capture the similarities. The semantic information lost from word co-occurrence count can be compensated if a model learns local contextual features from word ordering like IWE. As a result, the word embeddings, although trained in limited vocabulary, will be able to generate better QE terms that will help in increased retrieval efficiency in the task of answer passage retrieval.

- Cao and Lu [13] used the IWE model to learn word embeddings which performed better than embeddings trained by CBOW or Fastext (which takes into account character tri-gram but without the convolutional layer) in the word similarity and analogy tasks. Their ablation study showed that both the input representation and the convolutional layer helped in the tasks. Since obtaining QE terms is finding terms similar to our query, we believe that IWE model will also generate better QE terms than CBOW and thereby improve retrieval effectiveness.

- Cao and Lu [13] showed that IWE's performance is robust to fewer data. That is, in the lack of sufficient training data the performance of IWE in these tasks is not affected as much as CBOW. We will be training our word embedding models on candidate set of passages for the locally trained embedding models (query-specific corpora) whose vocabulary is significantly less than what Mikolov et al. [87] used to train CBOW (692K words) and Cao and Lu [13] used to train IWE (50K words). Hence, we believe that embeddings produced by IWE will capture more semantic relationships among words than CBOW in this data-scarce scenario which will lead to better QE terms generation and ultimately improved retrieval effectiveness.

- CNNs are simpler architectures than LSTMs or RNNs having a lesser number of parameters to train. CNNs are faster since they can be trained in parallel, unlike RNNs which require sequential computation [10].

3.2 Training the Word Embedding Models

The word embedding models - CBOW and IWE - are trained on a vocabulary/corpus to form the numerical representations of the words. In our thesis, we are interested to observe how retrieval effectiveness based on QE using these models change based on the data/corpus they are trained on for our task of answer passage retrieval. To that extent, we train the models on two kinds of data - one that is entire corpus-based (*global*) and the other that is specific (*local*) to a given query. In this context, let us define *target* corpus of the queries as the corpus that contains the relevant answer passages. We can use either the entire target corpus or another *external* corpus as the global corpus for training the word embeddings.

In the following sections we will discuss characteristics of each kind of data separately and the methodology we employ to train our word embedding models.

3.2.1 Training on a Global Corpus

A global corpus can be used to pre-train our word embedding models. These pre-trained embeddings can then be used to generate QE terms for answer passage retrieval. Various IR tasks ([124, 50, 152], etc.) pre-train these models on a global corpus or use publicly available pre-trained word embeddings as input to their IR model for tasks like document ranking. In our task, it makes the passage retrieval pipeline fast as we need to train the word embeddings once before we can generate QE terms for queries. The methodology is shown in Fig 3.6.

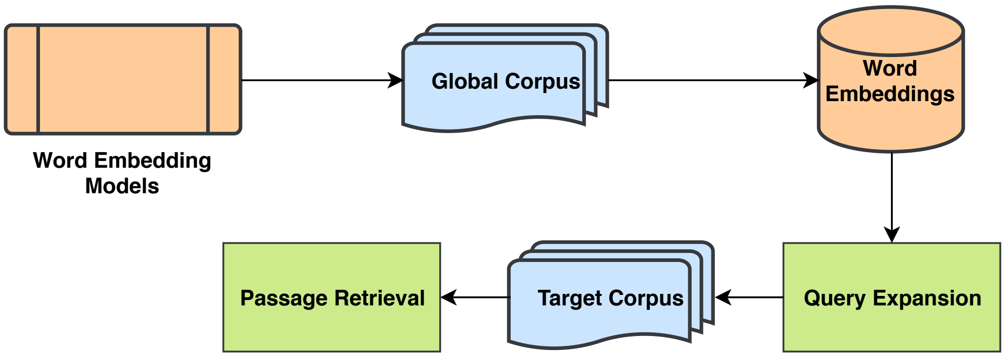


Figure 3.6: A pictorial overview of the retrieval pipeline with QE using word embeddings trained on a topically unconstrained corpus.

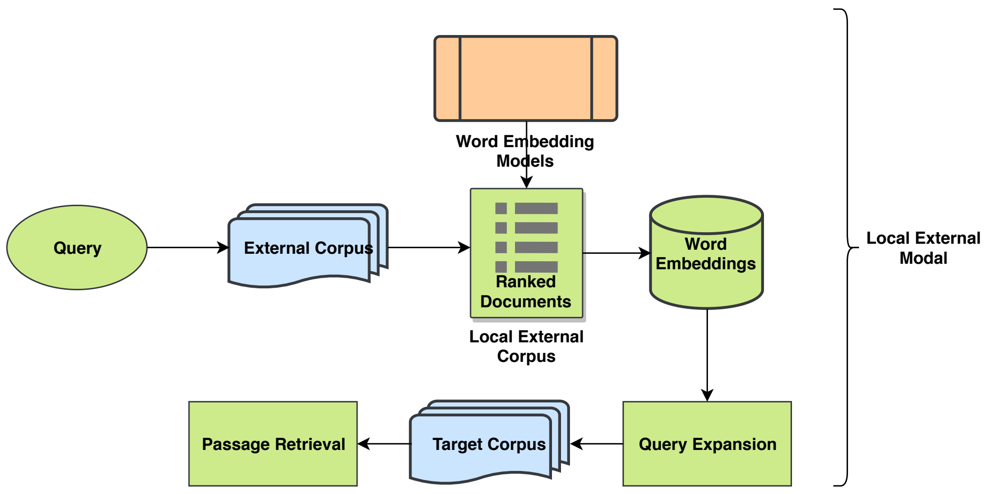
However Diaz et al. [37] pointed out that global word embeddings fail to capture topic-specific nuances of texts or documents. For example, the language statistics like word co-occurrence will be different in a document about ‘football’ than that in a document about ‘news’ or ‘law’. Thus word embeddings trained on a global general corpus will be different from those trained specifically in the document about ‘football’ or ‘news’. Diaz et al. [37] showed that word embeddings trained on these topic-specific corpora generate QE terms that increase the retrieval effectiveness of traditional IR models in the task of document ranking. This forms the inspiration to employ the same methodology in our task as well.

3.2.2 Training on a Local Corpus

As discussed, to retrieve answer passages to the question ‘Describe the game of football’ *we hypothesize that it is better to train our word embedding models on documents* *or passages about 'football' rather than an entire global corpus* (**H1**). To obtain such a query-specific or local corpora, we need to perform an initial round of retrieval for our query. The documents or passages obtained in such a round of retrieval are then used to train the word embedding models. Here, the assumption is that the language in these retrieved documents will be *consistent*, although all the documents might not be relevant to the query. Hence, word embeddings trained on these specific sets of documents will be better at capturing query-specific nuances consequently leading to better QE term generation. The topic-specific corpora can be obtained in two ways:

Local external: We can use an external corpus to perform the first round of retrieval and obtain a number of query-specific documents to train the models. The benefit of having an external corpus is that there is significantly more query-specific content of data for training the word embedding models. Diaz and Metzler [36] showed that an external corpus that is larger, high quality and comparable to the evaluation target corpus can help in IR tasks like document ranking. It can also provide sufficient topical diversity for producing good word embeddings. However, the disadvantages are since the training data comes from an external corpus they do not contain the answer passages and we need a large amount of data (around 1000 documents) to train our word embedding models. This makes the entire retrieval pipeline inefficient as reported by Diaz et al. [37].

Local target : We can also retrieve a specific set of passages/documents from the target dataset during the first round, which not only forms the dataset for training the local word embedding models but also forms the candidate set of passages to answer the given query. The benefit is that since we have less training data, training the word embedding models will be faster and consequently making the entire retrieval pipeline efficient. However, the disadvantage is the lack of words in the vocabulary might be problematic for the word embedding models. *We hypothesize that in this local target setting, where the models are trained on much less data than the local external or global model, IWE word embeddings will capture better semantic relationships and consequently generate better QE terms for passage retrieval than the CBOW model* (**H2**). The incorporation of both sub-word information and convolutional feature learning of context will contribute to better word embeddings for the IWE model when training data is less. A brief summary of the various corpus used to train our word embedding models is provided in Table 3.2. The local external and local target word embedding models are depicted in Fig 3.7. We will now look at the process of obtaining QE terms from this trained word embedding models.



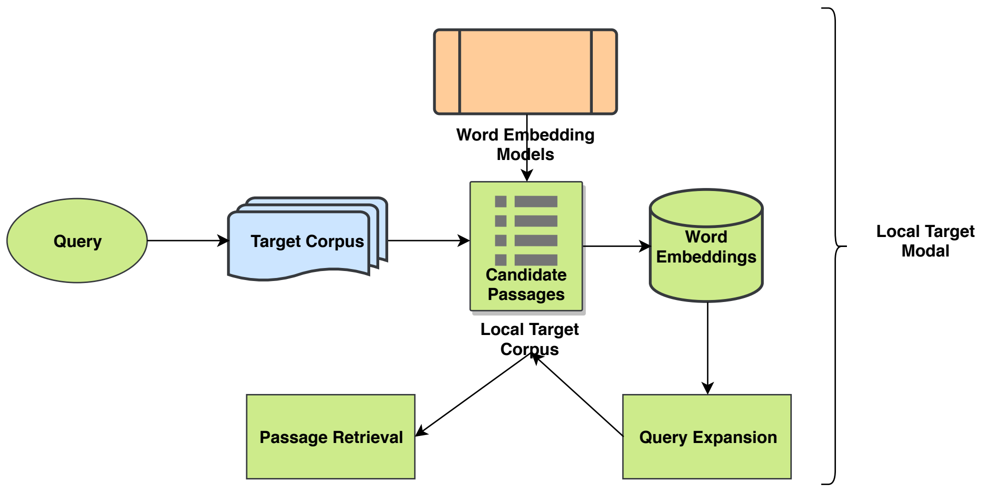


Figure 3.7: A pictorial overview of the retrieval pipeline with QE using word embeddings trained on a topic-specific corpus.

3.3 Query Expansion using Trained Embeddings

To generate QE terms from word embeddings we use the methodology of Diaz et al. [37]. Let $\mathbf{U}$ be the $|\mathbf{V}| \times d$ embedding matrix obtained from either globally or locally trained data (where $|\mathbf{V}|$ is the vocabulary size and d is the dimension of word embeddings) and q be the $|\mathbf{V}| \times 1$ binary encoding of the query. For example, if 'describe', 'game', and 'football' are the 1st, 2nd and 3rd words in the vocabulary, the question "Describe the game of football" will be represented by $q = [1, 1, 1, 0, \ldots 0]$. $\mathbf{S} = \frac{\mathbf{1}}{|\mathbf{U}|^2}\mathbf{U}\mathbf{U}^\mathbf{T}$ is a similarity matrix where element $\mathbf{S_{ij}}$ of the matrix is the cosine - similarity score of the ith and jth words in the vocabulary. $\mathbf{S}.q$ gives the weights of all candidate expansion terms to the given query. Out of which the top k terms are taken, normalized and used to form the expansion language model p_{q^+}. Fig. 3.8 depicts the process of obtaining QE terms from the trained word embeddings.

Table 3.2: Summary of corpus and training data for the word embedding models

Corpus	**Data**
External	Wikipedia dump
Target	`WikiPassageQA` (Introduced in section 3.4)

(a) Corpus used in our experiments

	Query Specific	**Training Data**
Global	No	Entire Wikipedia dump
Local External	Yes	Documents retrieved from Wikipedia dump to each query
Local Target	Yes	Candidate Passages to each query in `WikiPassageQA`

(b) Training data for the CBOW and IWE word embedding models

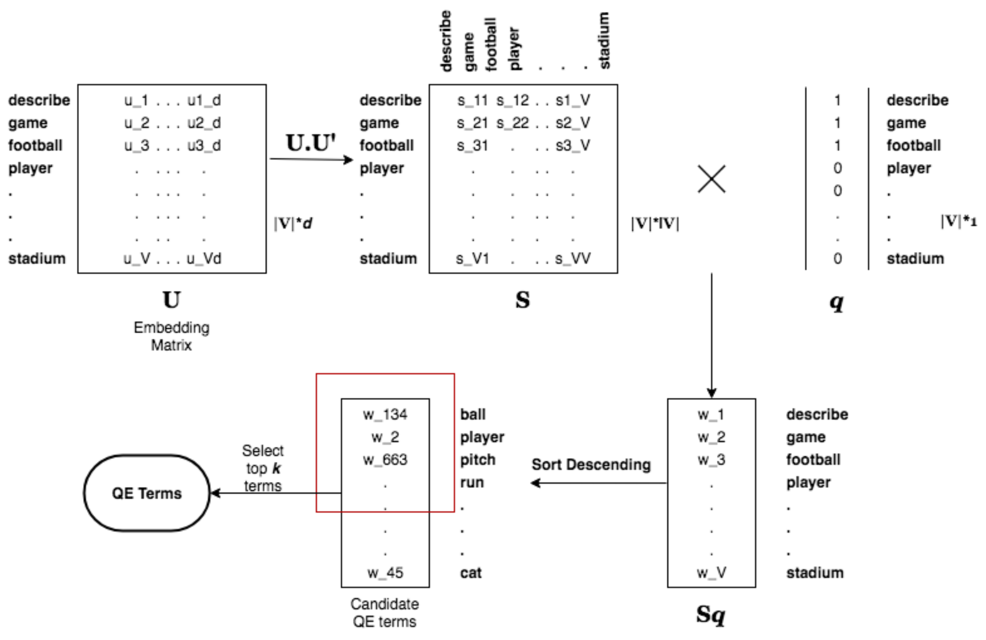


Figure 3.8: The QE using word embeddings methodology we use in our thesis

We interpolate p_{q^+} with the query language model $p_q(w) = \frac{tf(\mathbf{q}_i,\mathbf{q})}{|\mathbf{q}|}$ to form the final language model:

$$p_q^1(w) = \lambda p_q(w) + (1-\lambda) p_{q^+}(w) \tag{3.9}$$

where λ is the interpolation weight. This interpolated model is then used to rank passages from the candidate set of passages for a given query. The above equation is similar to that of PRF in Eq. 2.17 except that here the RM is replaced by the expansion language model obtained by finding cosine similarity of candidate terms with query terms. One of the important intuition of this approach is that candidate QE terms are ranked based on their similarity score will all query terms. Other works like [118, 68] often rank candidate QE terms based on their similarity score with individual query terms. But those approaches were evaluated on key-word searches instead of passage retrieval to natural language questions. We believe that the method adopted in our work is better suited when the queries are in the form of natural language. This is because such queries are typically longer ('football' vs 'Describe the game of football') and with the other approach QE terms can be influenced by just one or two query terms instead of them being semantically similar to the all the different important entities in the natural language queries.

3.4 `WikiPassageQA` Dataset

The `WikiPassageQA` dataset, has been developed by Cohen et. al. [24] for the task of *answer passage retrieval* for non-factoid questions. Given a query (in the form of a natural language question) and a Wikipedia document (in the form of a group of passages that make up the document), we have to rank the passages that contains the answer (*answer passages)* to the question above the ones that do not (*non-answer passages*). This can be seen as a non-factoid QA task where the answer is in the form of passages. There are over 4000 queries from the top 863 Wikipedia documents from the Open Wikipedia Ranking[2], making it "only large data set with long passages as answers for thousands of non-factoid questions in the open domain" [24] at the time of release (July 2018).

Each query in the `WikiPassageQA` dataset belongs to one unique Wikipedia document. The dataset comes with a binary passage-level relevance judgment - if a passage of the Wikipedia document contains the answer it is labeled as 1, otherwise 0. On an average, there are 1.7 *relevant* passages per question. Multiple questions can come from the same Wikipedia document albeit with different answer passages. In other words, each question has a set of candidate passages or passages that make up the respective Wikipedia document, out of which only a few are relevant (answer passages) for that question. As a result, *for our thesis we need not perform the first round of retrieval to obtain the candidate set of passages for our local target model.* We use the passages of the relevant Wikipedia document for the same. Table 3.4 provides two examples from the dataset.

The average vocabulary size of a Wikipedia document is 1752 unique words with a minimum of 84 words and a maximum of 4386 words. Figure 3.9 shows a distribution of the document length in the dataset. Each Wikipedia document in the `WikiPassageQA` dataset is split into passages of six sentences which was achieved using a non-overlapping sliding window of the same length. Hence, the last passage of the Wikipedia documents may contain less than six sentences. Employing this process on the 863 documents in the corpus yields 50,477 unique passages - each containing 135.2 words on average with a minimum of 11 and a maximum of 1332. The 4,186 questions in the dataset were created by crowd-workers from Amazon Mechanical Turk[3]. The length of questions varies from a minimum of 2 words to a maximum of

[2]See `http://wikirank.di.unimi.it/`.
[3]See `https://www.mturk.com/`.

39 words with an average length of 9.5 words. The relevance judgments were also obtained from the same crowd-workers that created the questions and were further validated by a subsequent mechanical turk verification poll. The dataset comes with a pre-defined training, development and testing set split respectively containing 3332, 417, and 416 queries which we maintain in our work.

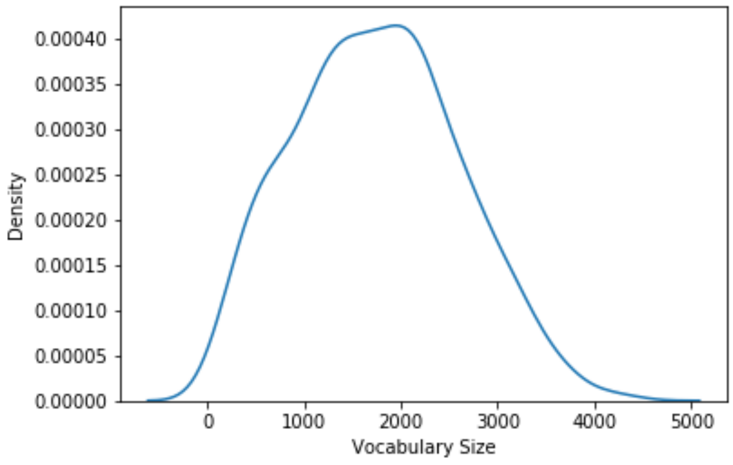


Figure 3.9: Distribution of vocabulary size of the 863 Wikipedia documents in `WikiPassageQA` dataset.

Table 3.3: An overview of `WikiPassageQA` dataset adapted from [24]. “P" in the first column denotes “Passages”.

Data	**Total**
Questions	4,154
CandidateP	243,489
PosCandidateP	6,947
NegCandidateP	236,542
PositiveP/CandidateP	0.03
CandidateP/Query	58.62
PsoCandidateP/Query	1.67
AvgLenOfQuestion	9.52
AvgLenOfAnswerP	146.8
AvgLenOfP	135.46

3.4.1 Evaluation Measures and Statistical Significance

Cohen et al. [24] reports the result of two baselines, three traditional IR models and five neural IR models on the dataset. None of their models, however, is a QE based model. Following their work, we use mean average precision (MAP), precision at k (P@10) and recall at k (Recall@10) metrics for our evaluation and comparison of re-

Table 3.4: Two examples from the `WikiPassageQA` dataset adapted from [24]. Each query comes from one Wikipedia document which is made up of a number of passages (candidate passages). The question can be answered by one or more passages out of the candidate set of passages. We have highlighted the passages containing answer with green and shown one example of a negative passage- that is a passage that does not contain the answer with red.

Query 1184:
Question: How is uranium found in nature?
Document ID: 745
Document Name: Uranium.html
Answer Passages:

Passage ID 1: It occurs naturally in low concentrations of a few parts per million in soil, rock, and water, and is commercially extracted from uranium-bearing minerals such as uraninite. In nature, uranium is found as uranium-238 , uranium-235 , and a very small amount of uranium-234. Uranium decays slowly by emitting an alpha particle. The half-life of uranium-238 is about 4.47 billion years and that of uranium-235 is 704 million years, making them useful in dating the age of the Earth. Many contemporary uses of uranium exploit its unique nuclear properties. Uranium-235 has the distinction of being the only naturally occurring fissile isotope.

Passage ID 5: Uranium metal has a very high density of 19.1 g/cm3, denser than lead , but slightly less dense than tungsten and gold. Uranium metal reacts with almost all non-metal elements and their compounds, with reactivity increasing with temperature. Hydrochloric and nitric acids dissolve uranium, but non-oxidizing acids other than hydrochloric acid attack the element very slowly. When finely divided, it can react with cold water; in air, uranium metal becomes coated with a dark layer of uranium oxide. Uranium in ores is extracted chemically and converted into uranium dioxide or other chemical forms usable in industry. Uranium-235 was the first isotope that was found to be fissile.

Query 2402:
Question: What is the structure of Australia's members of parliament?
Document ID: 400
Document Name: Member_of_parliament.html
Answer Passages:

Passage ID 0: A Member of Parliament is the representative of the voters to a parliament. In many countries with bicameral parliaments, this category includes specifically members of the lower house, as upper houses often have a different title. Members of parliament tend to form parliamentary groups with members of the same political party. The Westminster system is a democratic parliamentary system of government modeled after the politics of the United Kingdom. This term comes from the Palace of Westminster, the seat of the Parliament of the United Kingdom. A member of parliament is a member of the House of Representatives, the lower house of the Commonwealth parliament. Members may use "MP" after their names; "MHR" is not used, although it was used as a post-nominal in the past.

Passage ID 1: A member of the upper house of the Commonwealth parliament, the Senate, is known as a "Senator". In the Australian states of New South Wales, Victoria and South Australia, a Member of the Legislative Assembly or "lower house," may also use the post-nominal "MP." Members of the Legislative Council use the post-nominal "MLC." Members of the Jatiyo Sangshad, or National Assembly, are elected every five years and are referred to in English as members of Parliament. The assembly has directly elected 300 seats, and further 50 reserved selected seats for women. The Parliament of Canada consists of the monarch, the Senate, and the House of Commons trieval effectiveness of different models. We employed `trec_eval`[4] for the calculation of all measures of retrieval effectiveness in the sanity check and all our subsequent experiments. Wherever reported, the statistical significance of the observed difference in retrieval effectiveness has been performed using Wilcoxon signed-rank test[5] with Bonferroni correction and p-value < 0.05.

Precision [82] is the fraction of relevant documents out of all the documents retrieved by a retrieval model. It measures the system's ability to *only* to retrieve relevant documents. It can also be evaluated at a given cut-off rank, considering only the topmost results returned by the system. This measure is called precision at k or $P@k$ (for our thesis, we use $P@10$).

$$P = \frac{\text{\# relevant docs retrieved}}{\text{\# docs retrieved}} \quad (3.10)$$

$$P@10 = \frac{\text{\# relevant docs retrieved}}{\text{\# docs retrieved (=10)}} \quad (3.11)$$

Recall [82] is the fraction of relevant documents retrieved by a model out of all the relevant documents in the collection given the query. It measures the system's ability to retrieve *all* relevant documents. When recall is evaluated at a given cut-off rank it is called Recall at k or $R@k$ (for our thesis, we use $R@10$).

$$R = \frac{\text{\# relevant docs retrieved}}{\text{\# total relevant docs}} \quad (3.12)$$

$$R@10 = \frac{\text{\# relevant docs retrieved}}{\text{\# total relevant docs}} \quad (3.13)$$

Both precision and recall are set-based results and does not take into account the order of the retrieved list. We calculate the sum of the metric scores per query and take the mean by dividing with the total number of queries $|\mathbf{Q}|$.

Average Precision [82] is a metric that takes the order of the retrieved result into account. It also uses the number of relevant documents $\mathbf{R}$ and employs an indicator $rel(k)$ which is set as 1 if the item at rank k is a relevant document, 0 otherwise. This average is calculated over all relevant documents retrieved in the top n documents and the relevant documents not retrieved get a precision score of zero. This metric can be evaluated on a collection level by taking mean (dividing by $|\mathbf{Q}|$) of the scores obtained per query.

$$AveP = \frac{\sum_{k=1}^{n} P@k \cdot rel(k)}{\mathbf{R}} \quad (3.14)$$

$$MAP = \frac{1}{|\mathbf{Q}|} \sum_{q \in \mathbf{Q}} \frac{\sum_{k=1}^{n} P@k \cdot rel(k)}{\mathbf{R}} \quad (3.15)$$

[4]See https://trec.nist.gov/trec_eval/. We have used version 9.0.
[5]We use `scipy.stats.wilcoxon` method of Python.

3.5 Research Questions

In our thesis, we investigate the usefulness of query expansion using word embeddings for the task of answer passage retrieval. We aim to answer the following research questions:

RQ1 Does QE based on locally-trained word embeddings [37] improve the effectiveness of traditional retrieval model for the task of answer passage retrieval compared to globally-trained embeddings?

Specifically, we are interested in finding out whether word embedding models (CBOW and IWE) trained on query specific corpora produce better QE terms than when they are trained on a global corpus. Generated QE terms are considered better when they increase retrieval effectiveness in our task of answer passage retrieval. We also observe how the local and global expansion models perform in comparison with a no expansion baseline and also with the strong RM3 QE baseline. We conduct an error analysis, investigating the cases where the QE using word embedding models leads to a decrease in retrieval efficiency, which forms an important stepping stone to **RQ2**.

RQ2 Does QE based on IWE word embeddings improve the effectiveness of traditional retrieval models compared to CBOW word embeddings when the models are locally-trained on limited data?

In particular, how do the two components of IWE, the input of sub-word features and convolutional feature learning of context, contribute to the improved retrieval effectiveness over CBOW, if any? Moreover, we also observe how vocabulary size, query length, and query term frequency affect the retrieval effectiveness of QE using CBOW and IWE models. We leverage the QE terms produced by the two models not only to compare retrieval effectiveness but also to perform an extensive comparison of the word embeddings produced by the two models when trained on less training data.

Chapter 4

Experiments and Results

In this chapter, we first outline our experimental setup (section 4.1), then we proceed to answer our two research questions. First, we compare between QE using locally trained word embeddings vs that using globally trained word embeddings (**RQ1**, Section 4.2). Secondly, we compare QE using IWE and that using CBOW when they are trained using query-specific data (candidate set of passages) with limited vocabulary (**RQ2**, Section 4.3).

4.1 Experimental Setup

In this section, we first discuss the implementation details of the traditional IR models that we use as baselines for our passage retrieval task (Section 4.1.1). We follow it up by a discussion of the experimental pipeline used for passage retrieval (Section 4.1.2), a summary of adopted hyper-parameter values for each model (Section 4.1.2), and finally a sanity check of our baseline models against the benchmarking results reported in Cohen et al. [24] (Section 4.1.4).

4.1.1 Traditional Retrieval Models

We use QL with Dirichlet Smoothing (introduced in Section 2.1.2) for the no expansion model and RM3 (introduced in Section 2.3) as our traditional retrieval models. We employed the implementation in the open-source search engine `Indri` [133], available with Lemur toolkit[1]. `Indri` is compatible with various Operating systems[2] and is one of the most popular search engines among IR researchers being used by them for over a decade [157, 149]. We further used a variant of `Indri` RM3 (RM3+EC), implemented by Fernando Diaz[3], where the first round of retrieval is performed on an external corpus (the Wikipedia dump), similar to our local external model. The purpose of employing this model is to observe whether incorporating external vocabulary improves the performance of RM3 for our task and how it compares to the other QE models that we use in our experiments.

[1]See https://www.lemurproject.org/indri/. We have used Indri 5.13

[2]We use both Windows 10 and a Unix based OS for our experiments.

[3]See https://github.com/diazf/indri

4.1.2 Experimental Pipeline

An overview of the experimental pipeline we adopted for obtaining the retrieval effectiveness using various word embedding models is depicted in Fig 4.1 and is further elaborated in this section.

1. **Data Pre-processing and indexing**: We removed stopwords[4] from both the documents and queries of the `WikiPassageQA` dataset. We only kept alphanumeric characters and removed the rest to make it compatible with `Indri` toolkit and also to avoid noisy terms during the training of word embedding models. Lastly, we observe a number of special cases in the `WikiPassageQA` dataset. We list them in the following and mention our decisions with regards to them:

    - We remove questions with ID 4148, 1315 (train) and 4149 (dev) since they only contained the symbols '{}'.
    - Questions with ID 3731 (train) and 3732 (test) are the same - "How does the WTO function?" but with different passages as the answer. We remove these instances of ambiguous queries.
    - All candidate passages of QID 903, 3727, 3738, 3729, 3993 are answers. We keep them for our experiments.
    - Document ID 188 does not have any questions associated with it. We keep it in the corpus for indexing.

    From here on, for our experiments and remainder of the pipeline, we refer to `WikiPassageQA` as the filtered and pre-processed instance. We index the dataset using `Indri` retrieval toolkit.

2. **Word Embeddings:** We use three sources of training data (Section 3.2) for our word embedding models which are listed below:

    - **Global**: we train CBOW and IWE models on a Wikipedia dump of June 2015[5]. For CBOW, we use the implementation of `Gensim`[6], which is a Python library developed for the target audience of NLP and IR community. We have coded the IWE model from scratch which is available online[7] for reproducibility and future usage. Both the word embeddings were trained for 15 epochs.
    - **Local target**: CBOW and IWE models are trained on the candidate passages corresponding to each query from the `WikiPassageQA` dataset. Both IWe and CBOW are trained for 50 epochs.
    - **Local external**: CBOW is trained on the top 1000 retrieved documents from the Wikipedia dump to each query. In this case, CBOW is trained for 15 epochs as well. The initial round of retrieval on the Wikipedia dump is performed using `Indri` toolkit.

[4]We used the SMART stopword list. See https://www.lextek.com/manuals/onix/stopwords2.html.
[5]See https://dumps.wikimedia.org/.
[6]See https://pypi.org/project/gensim/.
[7]See https://github.com/roynirmal/IWE.

The training on global and local target corpus can be done in an offline manner (before passage retrieval) in our case, whereas that for local external corpus has to be performed in an online manner (during passage retrieval).

3. **Query Expansion:** Once the word embedding models are trained, we use Cosine Similarity (Eq. 2.5) to generate QE terms employing the methodology mentioned in section 3.3. For the embeddings trained on a global corpus, we only have one set of embeddings for all queries. For the word embeddings trained on local target corpus, we store the embeddings trained on the respective candidate passages to each query in a lookup table. Hence, each query is represented by a different set of word embeddings which is looked up during this step. For the embeddings trained on the local external corpus, the word embeddings to each query are also different but they are obtained after the first round of retrieval. Like the training of word embeddings, the QE process is also done in python.

4. **Expanded Query:** The expanded query LM (Eq. 3.9) is obtained according to the methodology described in Section 3.3 which is then converted into a 'query file' compatible with the `Indri` toolkit. In the query file, we also provide the IDs of the associated candidate passages.

5. **Passage Retrieval**: We performed passage ranking with `Indri` QL model with Dirichlet smoothing using the subprocess[8] module of python which produced the files with ranked passage list for each question.

6. **Retrieval Effectiveness**: We use `trec_eval` for measuring the retrieval effectiveness with the metrics mentioned in the previous section.

The pipeline essentially follows the Cranfield methodology defined in Section 2.1.3. The experimental pipeline for passage ranking using the baseline models of QL (with no expansion), RM3 and RM3+EC remain the same without steps 2, 3 and 4. We pass the necessary arguments[9] for `Indri` toolkit in the command line for obtaining the results file with the ranked passages.

[8]See `https://docs.python.org/2/library/subprocess.html`.

[9]See `https://www.lemurproject.org/doxygen/lemur/html/IndriParameters.html` for the comprehensive list of arguments for different retrieval models.

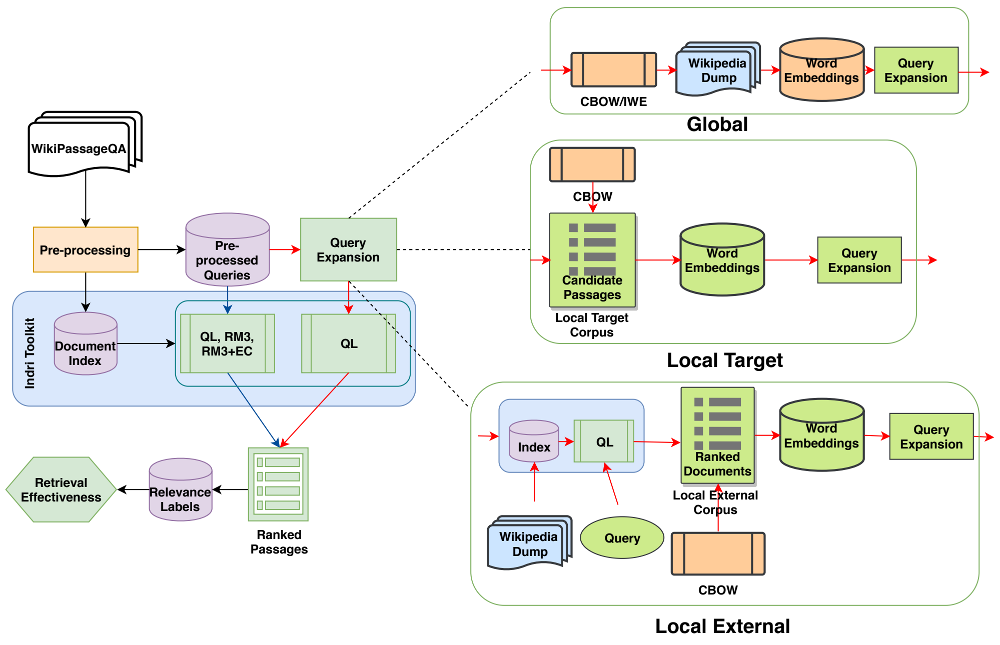


Figure 4.1: A block diagram of the pipeline. Green Blocks signify all processes that are done online during retrieval. Purple cylinders indicate everything that we can store offline. Red arrows indicate steps taken during retrieval using QE, blue arrows indicate retrieval using baseline models, and black arrows indicate steps common for both. The three word embedding training approaches for QE are shown.

4.1.3 Hyper-parameter Tuning

We tuned the hyper-parameters of the retrieval and word embedding models on the combined training and development set of the WikiPassageQA dataset, optimizing for MAP. We performed a grid search, which is one of the most widely used strategies for hyper-parameter optimisation in ML [8], over a range of values[10]. We have used the best performing combination of hyper-parameter settings for the experiments reported in our work which are summarized in Table 4.1.

For QE using word embedding models, the optimal dimension is lower when they are trained on limited vocabulary (local target) than when they are trained on larger vocabulary (global and local external). The optimal number of QE terms, on the other hand, is more for when the word embedding models are trained on query-specific corpus (local target and local external) than when they are trained on entire corpus (global). We have discussed a possible reason for this in Section 4.2.1.

4.1.4 Reproducing Benchmark Results

In order to ensure the retrieval models using the Indri toolkit is performing as expected we performed a sanity check using the QL model on the WikiPassageQA dataset. We compared it with the results as reported by Cohen et al. [24] and summarized it in Table 4.2. The difference in the result, which varies from −0.0267 to

[10]The range of hyper-parameter values we tried is provided in Appendix B.

Table 4.1: Final hyper-parameter values of (a) Traditional Models and (b) Word Embedding Models used for experiments on the test split of the `WikiPassageQA` dataset. Values marked with $*$ were not tuned. Models requiring two rounds of retrieval have two smoothing parameters - μ_1 and μ_2 respectively for each round.

Model	**Parameters**	**Tuned Value**
QL	μ	250
RM3	$fbMu$	750
	$fbDocs$	5
	$fbTerms$	400
	$fbOrigWeight$	0.5
	μ	250
RM3+EC	$fbMu$	250
	$fbDocs$	50
	$fbTerms$	200
	$fbOrigWeight$	0.4
	μ	500

(a) Traditional Models

Model	**Parameters**	**Tuned Value**		
		Global	Local Target	Local External
CBOW	μ_1	-	-	250^*
	α	0.01	0.1	0.01
	C	5^*	5^*	5^*
	N^-	100	30	30
	d	200	100	400
	k	100	200	200
	λ	0.45	0.6	0.3
	μ_2	500	500	500
IWE	α	0.02^*	0.02^*	-
	C	5^*	5^*	-
	γ	10^*	10^*	-
	l	3^*	3^*	-
	N^-	100	30	-
	d	300	100	-
	k	50	200	-
	λ	0.45	0.6	-
	μ	500	750	-

(b) Word Embedding Models

−0.0026, can be caused by the difference in pre-processing techniques of stemming and stopping (For example, we have not performed stemming). We consider the differences to be small and hence the model performs as expected.

Table 4.2: Sanity check of IR models by benchmark test on the `WikiPassageQA` as reported by Cohen et al. [24]. The difference of our obtained results with the original is mentioned in the last row.

	MAP	**P@5**	**P@10**	**R@5**	**R@10**
`WikiPassageQA` [24]	0.5436	0.1947	0.1151	0.6353	0.7275
QL	0.5282	0.1894	0.1125	0.6124	0.7157
Difference	−0.0154	−0.0053	−0.0026	−0.0229	−0.0118

4.2 RQ1: QE using Locally vs Globally Trained Embeddings

In this section, we first answer **RQ1** by comparing the retrieval effectiveness of QL no expansion model, QE using the RM3 model, and finally with QE using globally and locally trained QE models. We adopt the experimental pipeline as described in Section 4.1.2. The results are depicted in Table 4.3. Following that, we perform an **error analysis**, identifying 6 error types, where the QE using various word embedding models perform worse than the no expansion model. An overview of this section is provided in Fig. 4.2.

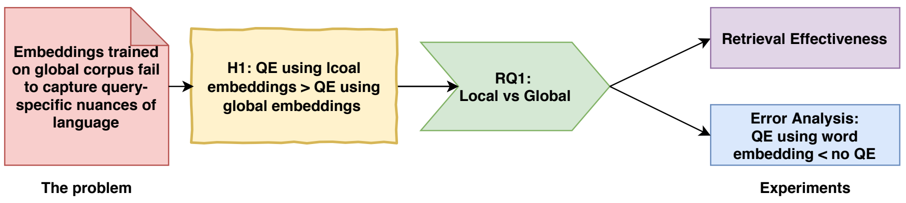

Embeddings trained on global corpus fail to capture query-specific nuances of language
The problem
H1: QE using lcoal embeddings > QE using global embeddings
RQ1: Local vs Global
Retrieval Effectiveness
Error Analysis: QE using word embedding < no QE
Experiments

Figure 4.2: An overview of the experiments and analysis done to answer **RQ1**

4.2.1 Comparing Retrieval Effectiveness

Firstly, as can be seen from Table 4.3, incorporating PRF using both RM3 (row 2) and RM3+EC (row 3) outperform the QL no expansion model. This is in line with the findings of [102, 68, 118] where QE using the RM3 model improves performance over the no expansion baseline in the ad-hoc search task. However, RM3 suffers from a drawback—it considers the top-ranked passages by the no expansion QL model as relevant and the number of optimal top-ranked passages being 5 (Table 4.1), the answer passages might not be present in those passages and consequently, the expansion terms generated will not help with retrieval effectiveness. Using the external Wikipedia dump to obtain the RM, proves to be better as RM3+EC significantly outperforms RM3. This shows that RM3 benefits from an external corpus to generate QE terms for passage retrieval by alleviating the above-mentioned problem of RM3.

We observe that both CBOW trained on local target data (row 6) and local external data (row 7) significantly improve performance over the no expansion baseline. However, the CBOW local target model does not have significant improvement over the CBOW trained on global data (row 4) or RM3 models. It is only when we train CBOW on a larger but query-specific external corpora (like the Wikipedia dump) that we see significantly improved performance over the baselines and QE using globally trained CBOW. This observation is on similar lines as observed by [37, 141] and can be attributed to the larger training data available when using the local external corpus. Here the training data for CBOW is diverse enough to obtain good QE terms. We observe that IWE trained on local target corpus (row 8) can significantly outperform QE using globally trained IWE (row 5) and the baselines of RM3 and no expansion QL. Hence, we do not require an external corpus of Wikipedia dump for IWE. This shows that word embeddings produced by IWE, even when trained by limited data produce QE terms that help to increase retrieval effectiveness.

Table 4.3: **RQ1**: Comparison of retrieval effectiveness of QE using locally-trained word embeddings vs globally-trained word embeddings and traditional IR baselines for CBOW and IWE. ‘ + EC’ denote when the first round of retrieval was performed on the external Wikipedia dump. The superscript numbers denote statistically significant improvements (Wilcoxon signed rank test with $p < 0.05$) with Bonferroni correction against the respective rows. * denotes a statistically significant improvement over approaches($^{1}–^{6}$). The best performing approach per metric is displayed in bold.

	Model	**MAP**	**P@10**	**Recall@10**
Traditional	^{1}QL	0.528	0.112	0.716
	^{2}RM3	0.533^{1}	0.113	0.72
	^{3}RM3+EC	$0.535^{1,2}$	$0.116^{1,2}$	$0.732^{1,2}$
Global QE	^{4}CBOW	$0.536^{1,2}$	$0.115^{1,2}$	$0.735^{1,2}$
	^{5}IWE	$0.539^{1,2}$	$0.116^{1,2}$	$0.735^{1,2}$
Local QE	^{6}CBOW	$0.540^{1,2}$	$0.117^{1,2}$	$0.743^{1,2,3}$
	^{7} CBOW+EC	0.545^{*}	$0.118^{1,2,3,4}$	$0.747^{1,2,3,4}$
	^{8}IWE	$\mathbf{0.560}^{*}$	$\mathbf{0.118}^{\mathbf{1,2,4,5,6}}$	$\mathbf{0.752}^{*}$

QE using globally trained CBOW and IWE models although outperforms the no expansion model and RM3, fail to do so in comparison to RM3+EC and CBOW+EC. One reason can be query drift, i.e. change in “intent” of information need between the original query and the expanded query [164]. QE using word embedding models trained on a global corpus leads to the generation of expansion terms which might not be present in the respective answer passages or the respective candidate set of passages leading to a change in the underlying intent of the original query. For example, for the query “How is cricket played?" the expansion terms generated by the global CBOW model were ‘football’, ‘players’, ‘game’, ‘match’ etc out of which terms like ‘football’ were neither present in the answer passages nor in the candidate passages and terms like ‘players’,‘game’, ‘match’ were common terms throughout all the candidate passages with low IDF score. It failed to capture important candidate QE terms that were present in the answer passages.

A general observation from both IWE and CBOW is that a major portion of QE terms obtained from embeddings trained on a global corpus are terms that have lower IDF because they are terms present throughout the candidate set of passages. These QE terms are non-discriminating and hence not useful for the task of passage retrieval. This is shown in Fig. 4.3 where we plot the distribution of the IDF score of the QE terms generated obtained from the different word embedding models. Locally trained word embedding models like CBOW local external and IWE local target as seen from Fig. 4.3 are better at identifying important, higher IDF terms as QE terms. CBOW trained on local target corpus is an exception, in this case, the reason for which we probe later in Section 4.3.4. The above-mentioned short-coming of globally trained word embeddings is also reflected by the fact that the optimum number of QE terms (k) for the global embedding models is less than the local embedding models (Table 4.1). More QE terms lead to more terms that are present throughout the candidate passages instead being very important terms from the answer passages.

We follow this result by an error analysis provided in Section 4.2.2 where we check the test cases where QE using word embedding models fail to outperform the no expansion model. Our goal is to obtain a better understanding of the CBOW model pertaining to its performance when it is trained on limited vocabulary. We aim to identify the shortcomings of the CBOW model and observe if IWE can overcome those.

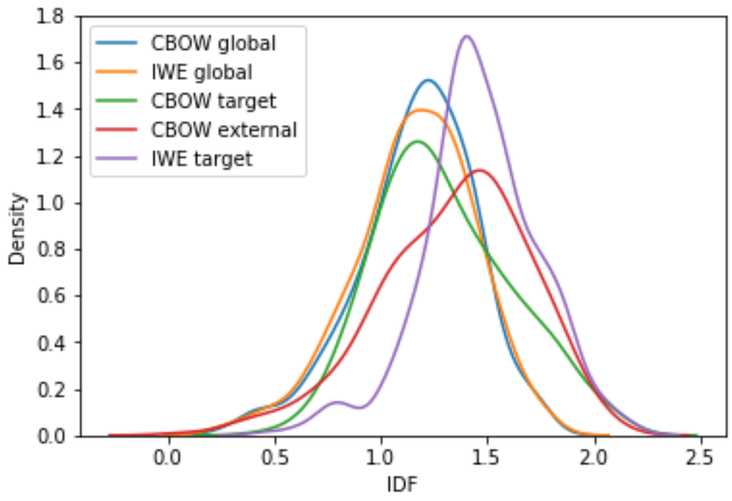


Figure 4.3: Distribution of the IDF value of QE terms generated by various models

To answer **RQ1**, *we state that QE using locally trained word embeddings significantly improves retrieval effectiveness of QL compared to QE using globally trained word embedding models and baselines models—no expansion and QE using RM3.*

4.2.2 Error Analysis

In this section, we first distinguish various types of terms present in a query and their importance with respect to obtaining QE terms and performing passage retrieval. We follow it by identifying 6 error types based on the type of query, answer passages and the generated QE terms in the cases where QE using word embeddings perform worse than the no expansion baseline.

Query term importance QE terms are terms similar to query terms in the embedding space. A query in the form of natural language questions is made up of a number of terms. The average length of a query in the `WikiPassageQA` dataset is 9.5 words. After stop-word removal from the questions, it is 5.8 words. Out of these words, some are important with relevant to the answer passages or the candidate passages (the Wikipedia document to which the query belongs). In this context, let's define the importance of a query term $\mathbf{q}_i$ in the query $\mathbf{q}$ with respect to the relevant answer passages $\mathbf{d}_{RP}$ as follows:

$$TI(\mathbf{q}_i) = \sum_{r \in \mathbf{d}_{RP}} tf(\mathbf{q}_i, r) * idf(\mathbf{q}_i, \mathbf{d}) \tag{4.1}$$

where $tf(\mathbf{q}_i, r)$ is the TF of $\mathbf{q}_i$ in the relevant passages r of the document $\mathbf{d}$ and $idf(\mathbf{q}, \mathbf{d})$ is its IDF in the whole document (in all candidate passages). That is, a query term is important for answering that question if its frequency in the answer passages is high and its IDF in the entire document is high as well (that is it is a discriminating term). For example, consider the query “What are the female aspects in blues from 1920s?" belong to the Wikipedia document titled “Blues". The query term ‘blues’ is highly frequent throughout the document and hence has a low IDF. Thus $TI(\text{blues})$ will be low although the TF of ‘blues’ in the answer passages is quite high. On the other hand, terms like ‘female’ and ‘1920s’ will have a high TI score if their frequency is high in the answer passages since their IDF score for the document is high as well. Figure 4.4 shows the distribution of TI scores for query terms in the test split of `WikiPassageQA` dataset. As we can see, most of the query terms have a TI score close to 0 and only a few terms in a query have a high TI score. Based on the distribution of the scores, we can divide the terms in a query according to the following:

- **Common**: When $idf(\mathbf{q}_i, \mathbf{d})$ is low i.e. when the query term is present in most candidate passages of the document and hence is non-discriminating. This will result in a low TI score for that term.
- **Rare**: When $tf(\mathbf{q}_i, r)$ is low but have a high $idf(\mathbf{q}_i, \mathbf{d})$ i.e. when the query term is not or rarely present in the answer passages but is a discriminating term in the document. Although the TI score will be low, the query term may contain important semantic information.
- **Important**: When both $tf(\mathbf{q}_i, r)$ and $idf(\mathbf{q}_i, \mathbf{d})$ are high, meaning the query term is rare and very important with respect to the answer passages. Query terms having a $TI > 0.02$ (75% ile) belong to these category.

As discussed in Chapter 3, QE terms are the terms in the embedding space with the top k average cosine similarity with all query terms. QE terms having very high similarity with one term can make it in the top k in spite of having low similarity with other terms. Ideally, we would want the QE terms to be highly similar to **important** query terms since these are the important terms that help to answer our question.

Error types Keeping these categorizations in mind, we manually check the entire test set where QE using word embeddings performed worse than the no expansion baseline. We identify 6 distinct error types based on the queries and QE terms generated. We calculated the percentage of occurrence of each error type for the word embedding models trained in different corpus and reported the numbers in Table 4.4. This experiment is inspired by a similar analysis done in [136]. The 6 error types are as follows:

1. **Complex Inference:** One of the most common types of error (as seen in Table 4.4) were to answer a question that requires complex reasoning. These are questions that are either long or characterized by the presence of **rare** query terms and the absence of **important** query terms. For example, in the question “How did Stalin’s death in 1953 affect the soviet union?" the TI score of none of the

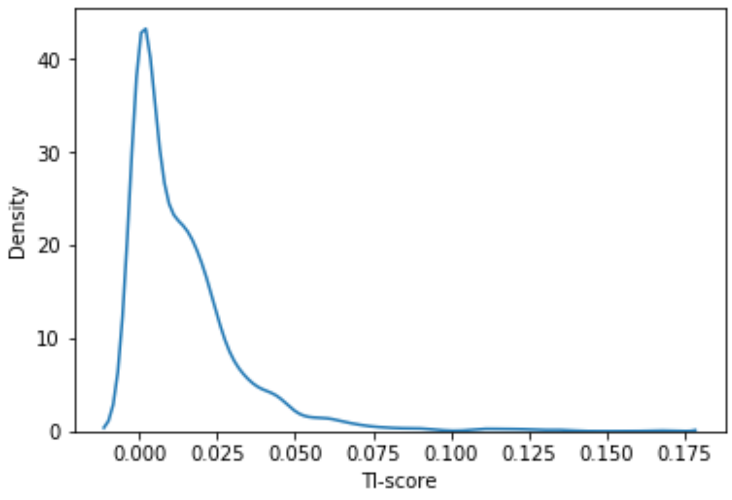


Figure 4.4: Distribution of TI scores for query terms in the test split of `WikiPassageQA` dataset

query terms was above 0.02. The term 'death' was present once in the answer passage and a few times in the Wikipedia document. Whereas 'affect' was not present in the answer passage but in one of the non-answer candidate passages. Hence, these were **rare** query terms. 'Stalin', 'soviet', 'union' were all **common** since the Wikipedia document is about Stalin. As mentioned, due to a lack of training examples, CBOW could not identify semantically similar terms to the important **rare** query terms in the answer passage.

2. **Missing Keyword** When a major question keyword is missing from the document, the local target CBOW fails to find any representation of that particular word. Without these important keywords, the question only consisted of **common** query terms. For example in the question, "How did Plato help instill the education process?" - the word 'instill' and 'help' were not present in the document but synonyms like 'established', 'founded' were. Without these key terms and stop-words, the remaining words like 'Plato', 'education' and 'process' are **common** query terms with very low $idf(t_q,d)$. Hence, none of the QE terms were related to 'instill' or 'help' but were terms similar to the present **common** query terms.

3. **Ambiguous Relevant Passage** This issue stems from how the `WikiPassageQA` dataset is created rather than the word embedding model's failure to generate good QE terms. Sometimes only one sentence in a candidate passage was part of the answer to the query. That passage was marked relevant in the dataset but it had mostly non-relevant content. For example, for the question 'How does the United States define an urban area?' the following passage was a relevant passage however only one sentence (highlighted) in the passage was a part of the answer:

The definition is an extent of at least 20 ha and at least 1,500 census

> residents. Separate areas are linked if less than 200 m apart. Included are transportation features. The UK has five Urban Areas with a population of over a million and a further sixty-nine with a population of over one hundred thousand In the United States, there are two categories of an urban area - the term urbanized area denotes an urban area of 50,000 or more people.

Although QE using word embeddings could retrieve other relevant passage(s), it failed to identify this passage as a relevant passage. Candidate passages like these are ambiguously relevant, as they are marked as answer passages in the `WikiPassageQA` dataset even though they are mostly made up of content non-relevant to the query. Hashemi et al. [56] also reports this issue of the `WikiPassageQA` dataset.

4. **Incorrect tokenization and non-English words:** The local embeddings failed to capture colloquial or non-English terms in the question/text like ‘tech’ (instead of technology), ‘ahimsa’ etc. This is similar to the **missing keyword** error except that the missing keywords are not English terms but either short forms or non-English terms. Without these keywords, these queries also consisted of only the **common** query terms.

5. **Wrong expansion terms generated:** In a few of the cases, the expansion terms generated were wrong or did not capture the underlying intention of the question and made the retrieval system to rank incorrect passages at the top.

6. **Unknown:** For approximately 5-10% of the error cases in different models, when none of the above error types applied, it was unclear why the retrieval effectiveness of local target CBOW was worse than global CBOW or the baseline.

From Table 4.4, we observe that **missing keywords** is the most important error type for CBOW trained on local target data. This is due to the fact that it is trained on limited vocabulary of the candidate passages and does not have embeddings for query terms that are not present in the candidate passages. This error is less prominent in case of models trained on global corpus or CBOW trained on local external corpus. It is less prominent for local target IWE as well, since as discussed in Section 3.1.2, IWE can form embeddings of unseen query terms, based on their sub-word information. Local target IWE also performs better than other models in the problem of **complex inference** error. As we will see in Section 4.3.3, local target IWE embeddings can generate better QE terms for longer questions. A general problem across models is the error of **ambiguous answer passage** which, as discussed, stems from the method of creation of `WikiPassageQA` dataset, rather than the QE models themselves.

Table 4.4: Comparison ($A < B$) of different pipeline variants for those topics (their number is shown in column #T) whose retrieval effectiveness is worse in A compared to B. For each topic, the percentage of each error type approach A (i.e. the worse performing one) is shown.

	#T	**Compl. inf.**	**Keyword missing**	**Amb. passage**	**Faulty tok.**	**System fault**	**Unknown**
Global CBOW < QL	100	24.0%	7.0%	30%	4.0%	23.0%	12.0%
Global IWE < QL	98	25.0%	8.0%	28.9%	5.4%	19.1%	13.6%
Local target CBOW < QL	95	29.5%	32.6%	16.8%	9.5%	6.3%	5.3%
Local external CBOW < QL	81	22.3%	8.8%	44.1%	2.3%	12.7%	9.8%
Local target IWE < QL	78	14.9%	5.0%	50.9%	6.4%	15.1%	8.6%

4.3 RQ2: QE using IWE vs CBOW

In this section, we first answer **RQ2** by comparing the retrieval effectiveness of QE using CBOW and IWE when they are trained on the local target corpus. Following that we delve deeper in order to understand the underlying reasons behind this difference in retrieval performance. Specifically, we observe the effect of vocabulary size or training data on the models (Section 4.3.2), effect of query length and query term frequency (section 4.3.3), followed by a closer look at the QE terms generated by the two models (Section 4.3.4), an analysis of the effect of two main components of the IWE model—sub-word information (Section 4.3.5) and convolution feature learning of the context (Section 4.3.6)—on the generation of QE terms and retrieval effectiveness. An overview of this section is provided in Fig. 4.5.

4.3.1 Comparing Retrieval Effectiveness

The main results of **RQ2** can be found in Table 4.5. IWE trained on the candidate passages of each query significantly outperforms CBOW trained on the same data. Its performance is similar to local external CBOW which is trained using more data. Hence, we can say that when trained on limited vocabulary IWE generates better QE terms and hence increased retrieval effectiveness than the QE terms obtained using the CBOW model. We observed that the IWE model improves over the CBOW model in about 36% of the total test cases. Out of those, about 60% of the improvements are in **complex inference** and **missing keyword** error types.

Thus, to answer **RQ2**, *we state that QE based on IWE word embeddings signifi-*

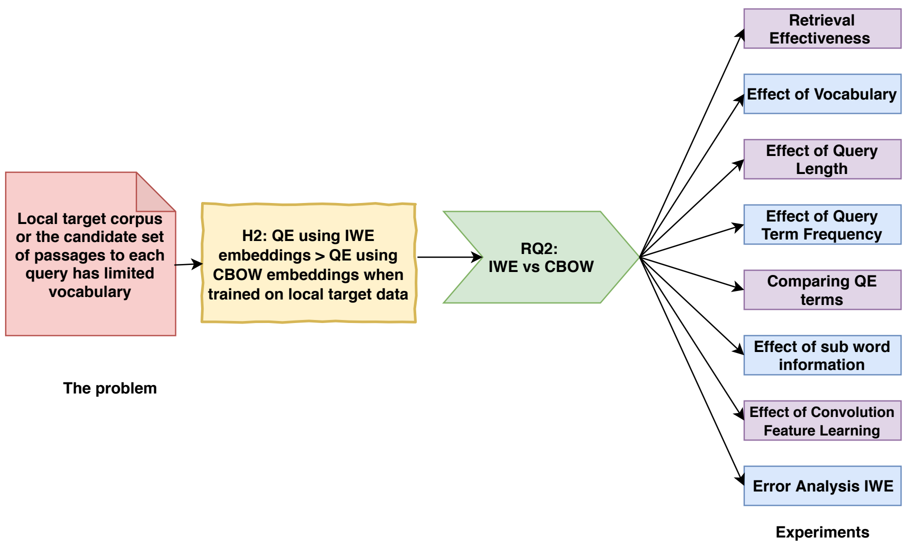

Local target corpus or the candidate set of passages to each query has limited vocabulary
The problem
H2: QE using IWE embeddings > QE using CBOW embeddings when trained on local target data
RQ2: IWE vs CBOW
Retrieval Effectiveness
Effect of Vocabulary
Effect of Query Length
Effect of Query Term Frequency
Comparing QE terms
Effect of sub word information
Effect of Convolution Feature Learning
Error Analysis IWE
Experiments

Figure 4.5: An overview of the experiments and analysis done to answer **RQ2**

Table 4.5: **RQ2**: Comparison of retrieval effectiveness of QE using locally trained CBOW vs locally trained IWE. The superscript numbers denote statistically significant improvements (Wilcoxon signed rank test with $p < 0.05$) with Bonferroni correction against the respective rows. The best performing approach per metric is displayed in bold.

	Model	MAP	P@10	Recall@10
Local QE	[1]CBOW	0.540	0.117	0.743
	[2] CBOW+EC	0.545[1]	0.118[1]	0.747[1]
	[3]IWE	**0.560**[1]	0.118[1]	**0.752[1]**

cantly improves the retrieval effectiveness of QL compared to that using CBOW when both the models are trained on the topic-specific candidate set of passages with a limited vocabulary.

4.3.2 Effect of Vocabulary Size

The candidate passages to each query, on average, consists of around 2000 unique words which, as we have seen, is orders of magnitude lesser than the vocabulary size word embedding models are generally trained on. We are interested to see if the size of training data in this range contributes to a difference in retrieval effectiveness. That is whether being trained using 1500 words or 3000 words has an effect on the quality of QE terms generated by the two models.

We divided the queries into four groups based on the distribution of the vocabulary size of their candidate passages (Fig. 3.9) which forms the training data for the word embedding models. For each vocabulary group, we observed the percentage of queries where QE using IWE resulted in better (>), worse (<) and equal (=) retrieval effectiveness than that using CBOW. In Fig. 4.6, we plot the percentage of queries in each vocabulary group where IWE $>$ CBOW, IWE $=$ CBOW and IWE $<$ CBOW. For all vocabulary groups, the number of queries where IWE $>$ CBOW is always higher than that where IWE $<$ CBOW. However, this difference is more prominent in the last two groups, where the vocabulary size is higher than average. This shows that, in our scenario of limited training data, there is not much difference in QE terms generated by IWE or CBOW when the vocabulary size is less than 1752 words (mean of the vocabulary size in the `WikiPassageQA` dataset). However, with slightly more training data (when the candidate passages contain more than 1700 unique words), QE terms obtained using IWE starts getting much better than that using CBOW which is reflected in the retrieval effectiveness. The improvement in the performance of IWE word embeddings with a larger vocabulary size can be attributed to its higher model complexity - a minimum amount of data is required to adequately reap the benefits of the IWE model.

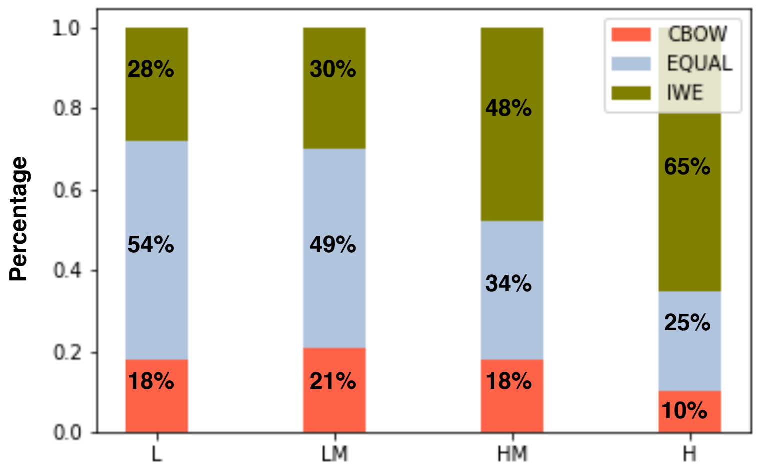


Figure 4.6: Effect of vocabulary size (training data) on QE terms generation by the word embedding models and consequently retrieval effectiveness. The vocabulary sizes are divided into the following groups - L: $v <= 1120$ (112 queries), LM: $1120 < v <= 1737$ (102 queries), HM: $1737 < v <= 2333$ (104 queries) and H: $2333 < v$ (98 queries). For every bar, the red, blue and green areas, respectively, depicts the percentage of queries for which CBOW performs better than IWE, both models have similar performance and IWE performs better than CBOW. One model performs better than the other if QE using that model results in higher retrieval effectiveness than that with the other model.

4.3.3 Effect of Query Length and Query Term Frequency

To observe the effect of query length and query term frequency in the candidate passages, we conducted two experiments inspired from similar experiments performed by Mitra et al. [91]. For the first experiment, we divided the queries based on the distribution of their length (in the number of words) and calculated the average MAP of the no expansion model, QE using CBOW and QE using IWE models in these groups (Fig. 4.7). In the second experiment, we divided the queries based on the frequency of the rarest query term in the candidate set of passages for that query. The frequency is calculated in the top x percentile - for example, a **common** query term will be in the top 5 percentile among the terms in the vocabulary. Similar to the previous experiment, we calculate the average MAP of the three models in each of these groups (Fig. 4.8).

As can be seen from Fig. 4.7, QE using IWE performs better than CBOW both in case of shorter queries and longer ($>$ 12 words) queries. Shorter queries usually consist of **common** query terms and QE terms obtained by CBOW embeddings are not as effective as that obtained using IWE embeddings. On the other hand, a question that requires complex inference is typically longer in length. As we have seen from Table 4.4, and further corroborated by Fig. 4.7, IWE embeddings are better obtaining QE terms for these long complex questions[11]. Interestingly for these longer queries QE terms produced by CBOW decreases the retrieval effectiveness of QL.

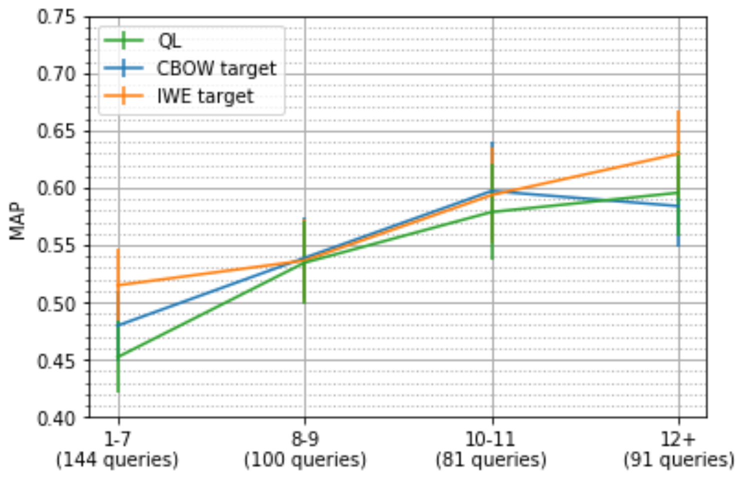


Figure 4.7: Relation between query length in words and retrieval effectiveness. Vertical lines denote standard error.

When the rarest query term belongs to the top 5% frequent terms of the vocabulary, then it indicates that most of the terms (if not all) in the query are **common** terms. As can be seen from Fig. 4.8, CBOW performs worse than the no expansion QL model in these cases. That is to say, when the query is made of highly frequent terms, the QE terms obtained by CBOW decreases the retrieval effectiveness of QL. We do not see such an effect on IWE. The reason for this observation is further explored in Section 4.3.4. When the query contains a term that is not present in the vocabulary, which forms the main reason for **missing keyword** error type, IWE performs better than CBOW. This is in line with the observation of Table 4.4 and can be attributed to the fact that IWE can form a representation of these unseen query terms from their tri-gram representations as seen in Chapter 3. CBOW can only achieve representations of unseen query terms when the first round of retrieval is performed from an external corpus and contains larger vocabulary. IWE does not need an external corpus (or a

[11]We have presented an exploratory analysis explaining the difference in QE terms obtained by IWE and CBOW word embeddings for longer queries in Appendix C

larger vocabulary) for finding a representation of words missing from the candidate passages. Thus, **missing keyword** error type is much less for IWE than CBOW when trained on local target corpus.

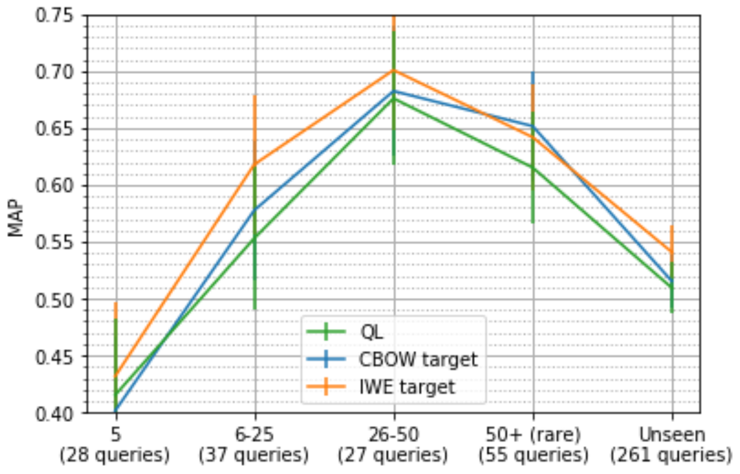


Figure 4.8: Relation between frequency (in top x percentile) of the *rarest* query term in the document and retrieval effectiveness. When the rarest query term is not present in the corresponding Wikipedia page we tag the query as 'unseen'.

4.3.4 Analysing QE Terms

In order to obtain insights into the word embeddings produced by CBOW and IWE trained on local target data, we look at the QE terms generated by the models. Word embeddings model like CBOW encode information regarding term frequency and the learned embeddings of rare words and popular words behave differently [47]. We argue that the effect of high-frequency words in a small corpus is much more prominent in the case of CBOW. We conduct the following experiment to see the influence of **common** query terms in QE terms generation in our task.

1. For each query, we select the top two **common** query terms - query terms with the lowest $idf(t_q, d)$.

2. We collect all words in their context window of ± 5 words which is the training window for CBOW and IWE.

3. Out of these collected words, we identify words that are rare in the candidate set of passages with a frequency of 1 or 2.

4. Lastly, we calculate what percentage of QE terms for the entire query is made up of these rare words in the context of high-frequency query terms. and plot the distribution in Fig. 4.9.

As observed from Fig. 4.9, a high percentage of QE terms in the case of CBOW are formed by the rare words which are present the context of high-frequency query terms. In the embedding spaces, these terms appear much closer to the **common** query

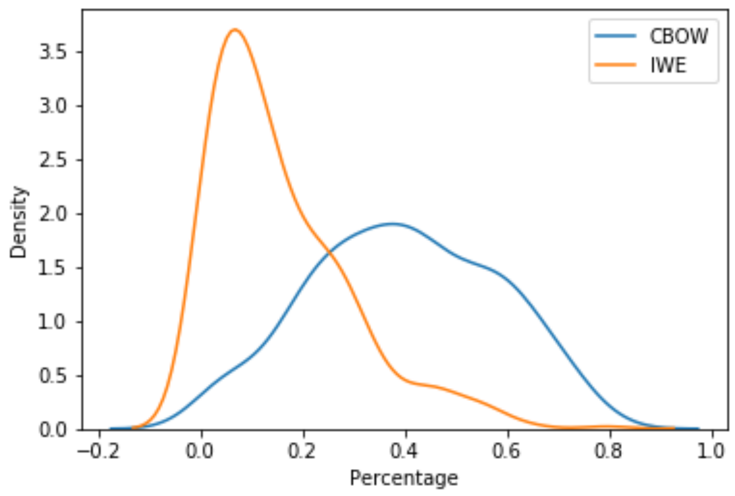


Figure 4.9: Distribution of the percentage of expansion terms that come from the immediate context of the low IDF terms.

terms although they are not semantically similar in most cases. As a result, these terms are selected as QE terms in spite of having no semantic relationship with the query. In most cases, especially in error types **complex inference** and **missing keywords**, these terms are present in one of the non-answer candidate passages, thus leading to a decrease in retrieval effectiveness. Only when these terms are present in the answer passages, there can be an increase in retrieval effectiveness. With IWE, we can see that the influence of the **common** query terms for QE term generation are lessened as the rare words in the context of these high-frequency terms makeup about 10% of the expansion terms on an average.

To qualitatively show the difference in word distribution in the embedding space we perform an experiment inspired by [37]. The assumption is that terms having high frequency (after removing stopwords) in the answer passages are good candidate QE terms. We can visualize where these good candidate QE terms lie in the embedding space of CBOW and IWE. We choose one of the questions where QE using IWE gives better retrieval effectiveness than CBOW (IWE $>$ CBOW) - “How did American football evolve originally?" from the Wikipedia document titled ‘American_football’[12]. Fig. 4.10 depicts a 2-dimensional projection using t-sne [80] of the query terms together with candidate expansion terms from the vocabulary. The query term representation by the blue dot is the mean of the 2D projection of the embeddings of all query terms. The good candidate expansion terms are highlighted in green. As can be seen, these terms cluster closer to the query in the IWE embedding space whereas in the CBOW embedding space they are scattered among the poor candidates. The red points are the rare terms that occur in the context of **common** query terms (‘American’ and ‘football’) but not in the answer passages. These terms if selected as QE terms will potentially lead to a decrease in retrieval effectiveness. As we can see, they are grouped close to the query embedding in CBOW and far away from the same in case of IWE.

[12]https://en.wikipedia.org/wiki/American_football

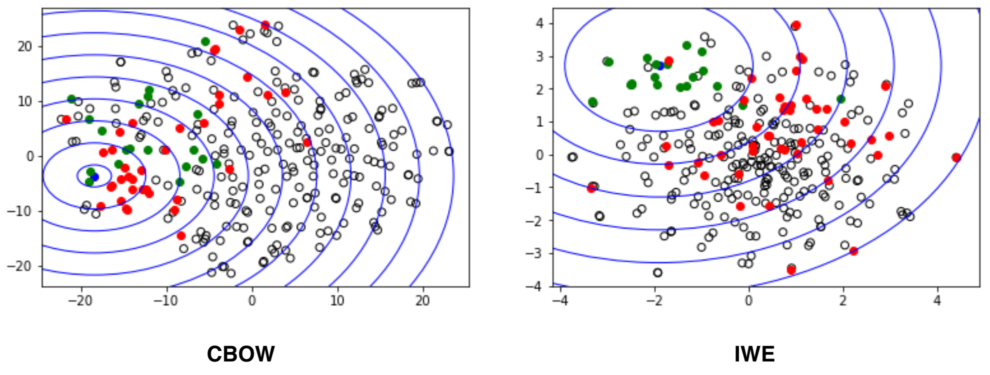


Figure 4.10: CBOW versus IWE embeddings of expansion terms. Each point represents a candidate expansion term. Green points have a high frequency in the relevant set of passages. White points have low or no frequency in the relevant set of passages. Red points are low-frequency terms in the vocabulary occurring in the context of the **Type1** query terms. The blue point represents the query 'How did American football evolve originally?' which is obtained by averaging the 2D representation of all the query terms. Contours indicate the distance from the query.

Thus, the green dots have higher chances of being picked up IWE as QE terms whereas the red dots have higher chances in case of CBOW. This can explain the better retrieval effectiveness for QE using IWE for this question. We observe a similar trend for most queries where IWE $>$ CBOW. This indicates a higher semantic similarity of good candidate expansion terms with the query in the IWE embedding space which perhaps leads to a better query expansion model and consequently better retrieval effectiveness. The projection of CBOW for our case is different from that presented in [37] were in their local model the good terms are clustered close to the query term embedding. This can be attributed to the limited vocabulary size on which our CBOW model is trained on and points to the fragility of the model in these cases.

In the following sections, we will perform ablation studies, understanding the effect of IWE model components on better expansion term generation and increased retrieval effectiveness.

4.3.5 Effect of Subword Information

One of the important components of the IWE model is the representation of the input words in their tri-gram form. The effect of this can already be seen from Table 4.4 where we see the IWE local model trained only on the candidate set of passages like the target CBOW model, brought down missing keyword error type to 5% from 32.6%. This is because, as mentioned in Chapter 3, the IWE model can have representations of unseen query terms when they are not present in the training data. In order to observe the effect of representing the input in such a manner on the better retrieval effectiveness of QE using the IWE model, we conducted the following experiments.

QE terms sharing similar sub-word information In this experiment, we aim to find out how many terms that are morphological variations of one of the query terms are captured as QE terms by the two word embedding models. We assume a term in the vocabulary to be a morphological variation of a query term if $>=$ 60% of the tri-grams of that term are tri-grams of one of the query terms as well. For example, terms like 'influences', 'influenced', 'influencing', 'influencer', 'male' etc. have $>=$ 60% of their tri-grams similar to that of one of the query terms for the query 'What were some aspects of female influence in the blues?'. Fig. 4.11 shows on average 10-15% of the QE terms picked up by CBOW embeddings are morphological variations of query terms. On the other hand that percentage is higher (around 20%) for IWE. This shows that one of the contributing factors for IWE $>$ CBOW is indeed the ability to identify morphological variant terms as QE terms.

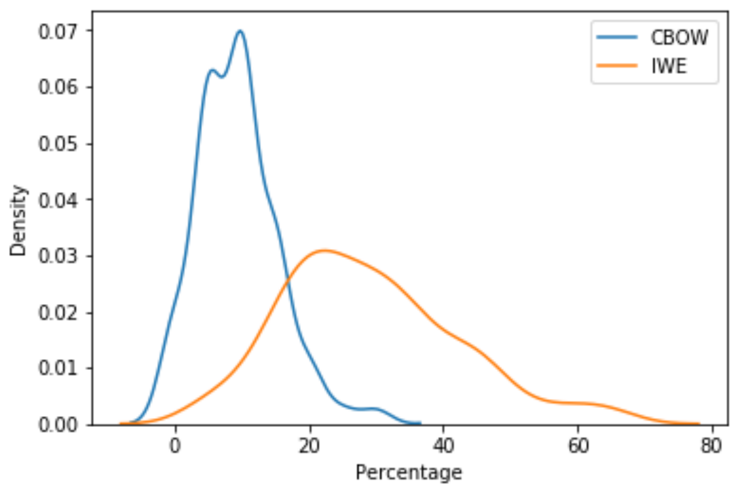


Figure 4.11: Distribution of percentage of expansion terms that share similar subword structure (trigrams) as one of the query terms

This raises the question of whether these QE terms sharing a similar tri-gram structure as the query terms are *solely* responsible for the performance improvement. In order to check that, we performed a retrieval where the QE terms were *only* formed by these terms that are a morphological variation of the query terms. The weights of the QE terms were computed with the same methodology as described in chapter 3. Table 4.6 shows that just the sub-word terms are not sufficient as QE terms and we require other semantically similar terms as QE terms for increased retrieval effectiveness.

Table 4.6: Comparing retrieval effectiveness of QE only using terms that are a morphological variation of the query terms with the local target CBOW and local target IWE models

	Model	**MAP**	**P@10**	**Recall@10**
Local QE	[1]CBOW	0.540[2]	0.117	0.743[2]
	[2]IWE (only subword)	0.471	0.117	**0**.739
	[3]IWE	**0.560***	**0.118**	**0.752***

Training CBOW with stemmed words Cao and Lu [13] do not perform stemming of the vocabulary while training their IWE model or for their experiments. Hence, in our thesis, we also do not perform stemming while training IWE or CBOW for a fair comparison. However, it raises another question whether CBOW with stemming will give similar performance as IWE since different morphological variations of a word will be normalized to the same word. In which case, we expect CBOW to obtain QE terms from answer passages that contain morphological variations of a query term. For this purpose, we stemmed the `WikiPassageQA` dataset using a Porter stemmer[13], indexed it and then proceeded to train our CBOW word embedding on the stemmed candidate passages for each query. Table 4.7 reports the retrieval effectiveness of QE using CBOW trained on the stemmed data (CBOW + stem). Although the performance of QE using CBOW increases with stemming, IWE still significantly outperforms both the CBOW models. This shows that IWE can capture sub-word information from the tri-gram representation and its usefulness is not merely a substitute for stemming. Perhaps, the filters of the convolution layer of the IWE model is learning various local features from the sub-word information alongside word order - a discussion on which is presented in the next section.

Table 4.7: Comparing retrieval effectiveness of QE using CBOW trained on the stemmed vocabulary to that using IWE trained on the original vocabulary of the local target corpus.

	Model	MAP	P@10	Recall@10
Local QE	^{1}CBOW	0.540	0.117	0.743
	^{2} CBOW + stem	0.546^1	0.118	0.747^1
	^{3}IWE	**0.560***	**0.118**	**0.752***

4.3.6 Effect of Convolutional Feature Learning of Context

The convolutional layer of IWE learns structural information about the context while predicting the target word during the training of word embeddings. That is to say, IWE takes the ordering of the context words while predicting the target word. Whereas, the CBOW model architecture has no such provisions. In order to observe how convolution feature learning of the context affects the word embeddings and consequently QE, we perform an experiment where we randomize the order of the words in the context of the target word while training both CBOW and IWE. Table 4.8 shows that the performance of QE using CBOW with random word order does not change significantly. This is due to the fact that while predicting the target word CBOW simply does a linear transformation where the order does not matter. On the other hand, we see that the performance of IWE decreases significantly when the word orders are randomized. This shows that the CNN layer of IWE learns local contextual features which are captured in the word embeddings and reflected in the retrieval effectiveness using QE. Randomizing the word orders, decreased the quality of QE terms produced by IWE.

[13]See https://tartarus.org/martin/PorterStemmer/.

However, the performance does not drop below that of CBOW, which shows that word co-occurrence[14] also has its contribution on IWE embeddings.

Table 4.8: Comparing retrieval effectiveness of QE using IWE and CBOW embeddings trained on local target data with (+ rand) and without randomizing the words in the context.

	Model	MAP	P@10	Recall@10
Local QE	[1]CBOW	0.540	0.117	0.743
	[2] CBOW + rand	0.541	0.117	0.735
	[3]IWE	**0.560***	**0.118**	**0.752***
	[4]IWE + rand	0.547	0.117	0.74

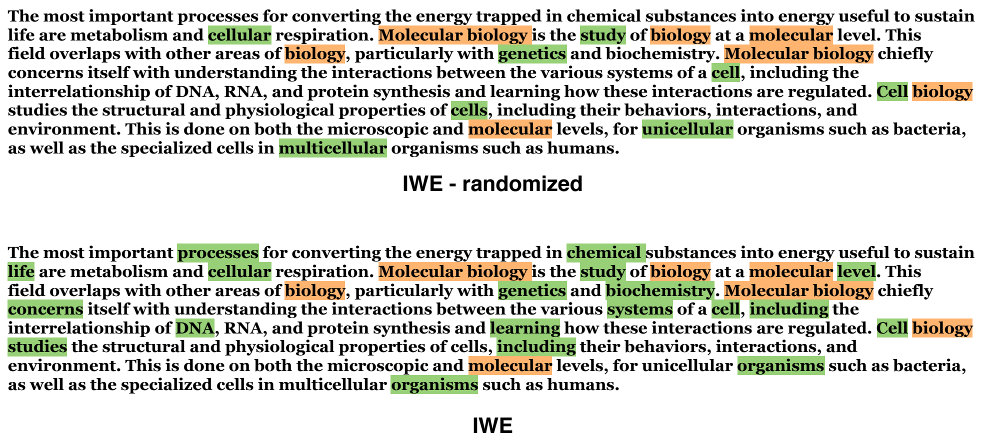
In order to understand what is being learned in the CNN layer while producing the word embeddings and how it is beneficial for the passage retrieval task, we check some of the test examples where the MAP drops the most (≥ 0.2 from IWE to IWE + rand). One of those cases is the question 'How would you describe molecular biology?' with QID 2821. Fig. 4.12 shows the QE terms generated by the IWE model with the randomized word order in a sentence and by the original IWE model. As we can see, when words are randomized in a sentence, IWE word embeddings fail to capture semantic relationships. Simple co-occurrence based information is not enough as the vocabulary size is small in the candidate set of passages. Hence, we can say that IWE learns important contextual information from local word ordering to predict word embeddings which capture various semantic relationships even with scarce training data, which is typically not possible by CBOW.

4.3.7 Error Analysis

We also analyzed the cases when the IWE model was performing worse than the CBOW model and saw that one of its strength of being able to use sub-word information for better expansion terms generation was also a cause for decreased retrieval efficiency in some cases. About 50% of the cases where it was worse than CBOW was due to the generation of wrong expansion terms since this model has an affinity of picking up terms that have a similar sub-word structure to a query word and those expansion terms made the system retrieve wrong passages.

Examining a few of such examples, we noticed that wrong expansion terms were typically generated when terms within the query had low or negative cosine similarity to each other. In these cases, if an **important** query term had low cosine similarity to its similar terms than a non-important term (**common**), it led to the wrong candidate expansion terms having higher weights, inevitably leading to decrease in retrieval performance. For example, in the query 'What were some aspects of female influence in the blues?' - 'female' is an **important** query term. However, it had very low cosine similarity with other terms in the query like 'aspects' and 'influence'. This means that terms that are similar to the other query terms are unlikely to be similar to 'female' and vice versa. Since we are taking into consideration all query terms together for

[14]Word co-occurrence count does not change by randomizing word order in a sentence


The most important processes for converting the energy trapped in chemical substances into energy useful to sustain life are metabolism and cellular respiration. Molecular biology is the study of biology at a molecular level. This field overlaps with other areas of biology, particularly with genetics and biochemistry. Molecular biology chiefly concerns itself with understanding the interactions between the various systems of a cell, including the interrelationship of DNA, RNA, and protein synthesis and learning how these interactions are regulated. Cell biology studies the structural and physiological properties of cells, including their behaviors, interactions, and environment. This is done on both the microscopic and molecular levels, for unicellular organisms such as bacteria, as well as the specialized cells in multicellular organisms such as humans.

IWE - randomized

The most important processes for converting the energy trapped in chemical substances into energy useful to sustain life are metabolism and cellular respiration. Molecular biology is the study of biology at a molecular level. This field overlaps with other areas of biology, particularly with genetics and biochemistry. Molecular biology chiefly concerns itself with understanding the interactions between the various systems of a cell, including the interrelationship of DNA, RNA, and protein synthesis and learning how these interactions are regulated. Cell biology studies the structural and physiological properties of cells, including their behaviors, interactions, and environment. This is done on both the microscopic and molecular levels, for unicellular organisms such as bacteria, as well as the specialized cells in multicellular organisms such as humans.

IWE

Figure 4.12: QE terms generated by the IWE model with the randomized word order in a sentence and by the original IWE model to the question 'How would you describe molecular biology?'. Terms highlighted using orange are the query terms and that using green are the QE terms generated by the IWE model that is presented in the answer passages out of the top 30 QE terms. Words semantically similar to query words like 'concerns', 'studies', 'learning', 'dna', 'biochemistry' etc are identified as QE terms by IWE. When the words are randomized, the convolutional layer of IWE fails to capture semantic information from the context and hence these terms do not appear as QE terms. Only terms sharing similar sub-word information are captured when word orders are randomized.

the generation of QE terms, terms similar to **important** query term terms might be given low weight in the expansion model. The QE terms generated for this query were {influences, influenced, blues, male, influencing, buyers,..}. Out of this only 'male' and 'buyers' were the terms closest to 'female' and were the only expansion terms to be present in the answer passage—these were good expansion terms. But since they had very low similarity with other query terms they were pushed down the list of candidate expansion terms leading to a decrease in performance. This phenomenon of query terms having low similarity with each other is what we define as **inter query disagreement**. We followed it by an experiment to observe if this was a general trend among the cases were the IWE model performed worse than the CBOW model (IWE $<$ CBOW).

We calculated the inter query term similarity for all terms within a query separately for the cases where IWE $>$ CBOW and IWE $<$ CBOW. Then we took the difference of the highest cosine similarity and lowest cosine similarity for each query as a measure of **inter query disagreement**[15]. We expect a higher inter query disagreement for cases when the IWE model performed worse than the CBOW model IWE $<$ CBOW. This is supported by the results obtained in Fig. 4.13 where we plot the distribution of inter query disagreement scores for both the cases. We believe that this problem is more pertinent in the task of QE for question answering rather than in key-word based

[15]Here we omit 12 cases in total where the number of query terms after stopping was reduced to 1

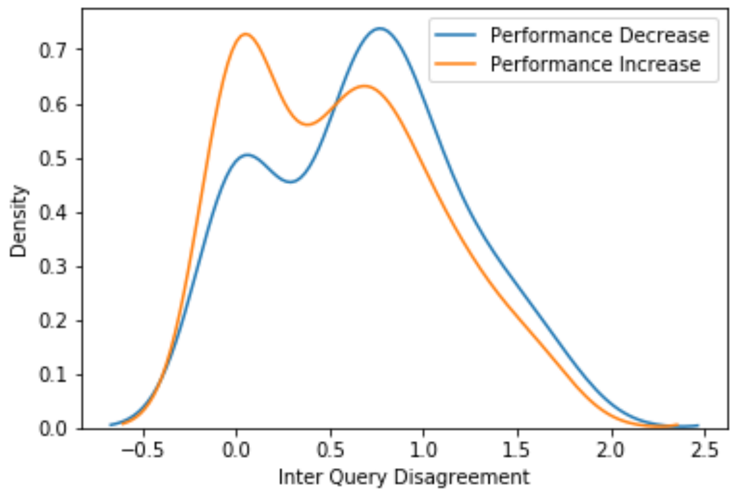


Figure 4.13: Distribution of **inter query disagreement** score for the two cases where the IWE model performed better and worse than the CBOW model. The inter query disagreement score is the difference between the highest term similarity and the lowest term similarity within a particular query

information retrieval. This is because questions are typically longer and the expansion model has to take into account more query terms while generating candidate QE terms. This motivates a different strategy for picking up QE terms. Instead of taking the mean cosine similarity of candidate terms with the query terms, we can weigh them using their IDF score or other measures. We leave this as future work.

Chapter 5

Conclusions and Future Work

In this chapter, we first draw conclusions (Section 5.1) of our work to answer our research questions and address the current research gap. Finally, we propose various future directions of research (Section 5.2) in consideration with the limitations of our work.

5.1 Conclusion

Since Mikolov et al. [85] proposed `word2vec`, word embeddings have been extensively used in IR and NLP community for a number of downstream tasks. In this thesis, we employ word embeddings on one such IR task—passage retrieval for open-domain QA using QE. We re-implemented the retrieval pipeline of Diaz et al. [37] in this new task.

QE has been shown to increase the retrieval effectiveness of traditional term-matching based IR models in the ad-hoc search task. However, *the effectiveness QE in the task of passage retrieval to open domain QA is relatively unexplored*. Employing the pipeline of Diaz et al. [37] not only allowed us to address the above research gap, but we could also observe *to what extent training the word embedding models on query-specific corpora improve the retrieval effectiveness in comparison to training them on the entire corpus*. Our experiments showed that QE using word embedding models improve the retrieval effectiveness of traditional IR models in this new task as well. Moreover, we found out that the retrieval pipeline of Diaz et al. [37] is generalizable to this new context—QE using embedding models trained on query-specific corpus improves the retrieval effectiveness of the traditional IR models in comparison to training them on the entire corpus (**RQ1**). We found out that, the QE terms generated by training the models on a global corpus were non-discriminating with respect to the candidate passages of the query which can explain the observed results. We conducted an extensive error analysis observing individual cases where QE using word embedding models failed to outperform the no expansion baseline. We identified 6 error types based on the terms present in the query and the candidate passages.

Next to observing the effectiveness of training word embedding models on query-specific data, we have researched the quality of word embeddings when the two models—CBOW and IWE—were trained on limited vocabulary. Despite a large number of works that have used popular word embeddings models or proposed their own word embeddings models for various tasks, *there is a lack of extensive studies showing the* *effect of limited vocabulary for training these word embedding models*. We found that when trained on less data IWE word embeddings contain more semantic information than CBOW word embeddings. This was shown by the fact that QE terms generated by IWE word embeddings increased the retrieval effectiveness of QL more than the QE terms by CBOW word embeddings—the former QE terms were also semantically more similar to the query than latter (**RQ2**). We found that rare terms in the vocabulary lie very close to high-frequency terms in the CBOW embedding space even though they are not semantically similar. IWE is immune to such drawbacks. Moreover, IWE can form representations of unseen query words using sub-word information unlike CBOW. We also observed that IWE embeddings produced better QE terms when the questions were longer. IWE gains its advantage over CBOW owing to the two components—incorporation of sub-word information and convolutional feature learning of the context— which contribute to the improved performance of IWE word embeddings.

Finally, we showed evidence that both of these components are able to capture important semantic features to learn the word embeddings. Representing the input using sub-word information helps the model capture the semantic information present in the morphological variations of a word. As a result, IWE embeddings can capture words having a similar sub-word structure to the query terms as QE terms. We also found evidence that the convolutional layer of the IWE model is learning important semantic information from word order in a sentence while producing word embeddings.

We would also like to comment on the reproducibility of the work of Diaz et al. [37]. The authors described a precise pipeline and clear statements in addition to mentioning all the hyper-parameter they optimized. This enabled us to accurately recreate their retrieval pipeline. Although we train our word embedding models on the global corpus of Wikipedia dump, the pre-trained word embeddings[1] they use for the global model are publicly available. They employ a Wikipedia dump of December 2014 and the Gigaword corpus[2] as the local external corpus. However, the latter, which is responsible for the improved performance in their experiments, is expensive and would not have been possible for us to use for our work.

Concluding, we corroborate the findings of Diaz et al. [37] in our task of answer passage retrieval to open-domain questions. Our thesis provides further proof of the benefits of locally trained word embeddings which we believe will also be beneficial for other NLP and IR tasks. We hope that our work provides enough motivation for future researchers to explore the effect of local contextualized information in these tasks. Moreover, we use QE to gain insights on the word embedding models of CBOW and IWE and understand the difference when they are trained on limited vocabulary. We expect our thesis to be an informative resource for future work deliberating on the type of word embedding model to use in their tasks.

[1]They use four GloVe embeddings of different dimensionality trained on the union of Wikipedia and Gigaword documents. See http://nlp.stanford.edu/data/glove.6B.zip. They also use a word2vec embedding trained on Google News documents. See https://code.google.com/archive/p/word2vec/. Lastly, they use a GloVe embedding trained on Common Crawl data. See http://nlp.stanford.edu/data/glove.840B.300d.zip.

[2]See https://catalog.ldc.upenn.edu/LDC2011T07.

5.2 Future Work

The effectiveness of QE using locally trained word embedding models pave the way for a number of research directions which we discuss in the subsequent sections.

5.2.1 Limitations

Our work has a few limitations that we would like to discuss and suggest improvements to address them.

Candidate set of passages In our thesis, the local target corpus is the candidate set of passages to each query. In the `WikiPassageQA` dataset, the candidate set of passages is already pre-defined. Thus we did not perform an initial round of retrieval to obtain the candidate passages. In many practical applications, the candidate passages might not be known a-priori. Hence, it would be interesting to see the effect of obtaining our own candidate passages and then training our word embedding models for QE. This kind of training data will be more noisy, containing more non-relevant information than having all candidate passages from the same Wikipedia document (as in the case of `WikiPassageQA` dataset). `MSMarco` (MicroSoft MAchine Reading COmprehension) released by Nguyen et al. [98] can be used for this improvement. Although the authors have made 1000 candidate passages available to each question, these passages do not come from a single document—they are snippets extracted from real web documents through Bing[3]. Moreover, the authors suggest researchers to use their own top-k retrieval model to obtain the candidate set of passages in response to each query.

Efficient retrieval pipeline Training the word embeddings locally requires an initial round of retrieval if the query-specific corpus is not known a-priori (in `WikiPassageQA` each query comes with an associated candidate set of passages). This makes the local QE embedding models slower than QE using word embedding models trained on a global corpus. This is short-coming is also pointed out by Diaz et al. [37]. The retrieval effectiveness obtained can be nullified by the decrease in time efficiency in many practical applications. Hence, one important future work will be to increase the efficiency of the QE using local word embedding models. If the entire corpus is pre-divided into coarser topics, instead of query-specific corpora, word embeddings can be trained on them in an offline manner—similar to when we know the candidate set of passages to a query beforehand. During retrieval time, the word embeddings trained on the respective rough sub-topics can be looked up to obtain QE terms.

Hyper-parameter tuning In our thesis, we have not fully explored the effect of all hyper-parameters for the IWE model. For example, as reported in Table 4.1, we fixed the value of learning rate (α), the number of context words (C) and temperature coefficient (γ) to values as suggested by Cao and Lu [13]. Moreover, we have not compared the robustness of the two models, IWE and CBOW, with respect to variations in their hyper-parameters. It would be interesting to observe the variation of retrieval effectiveness using QE when the model hyper-parameters are also changed. Depending

[3]See https://www.bing.com/.

on the robustness of the model we can strengthen or relax conclusions drawn in our thesis regarding the retrieval effectiveness. Word embeddings are known to have stability issues [58]—retraining the word embeddings with exact same hyper-parameters will lead to different embeddings. Hence understanding the effect of hyper-parameters will also aid in the reproducibility of our experiments and observations. In this regard, we can take inspiration from a number of literature [108, 40, 15] that aims to understand the impact of various model hyper-parameters on word embeddings.

Convolutional learning There is scope to delve deeper into understanding what precisely is being learned in the convolutional layer while producing the word embeddings. We have shown with one experiment (Fig. 4.12) how randomizing word orders in a sentence leads to word embeddings without semantic information—QE terms generated by those embeddings are not semantically related to the query terms for that particular example. This shows that the CNN layer is able to capture semantic information from the word order in a sentence. Additional empirical analysis is required to understand what is specifically learned and how are they learned—whether the filters or the max-pooling operation or both play a part in the semantic understanding. Shen et al. [126] has shown, using a few examples, that semantically similar words, even though they are not present in the same context, activate the same filters for the task of estimating document title relevance to a query. A study can be conducted to observe if this observation holds while predicting a target word from the context words—if semantically similar target words are activated by similar context representation (Section 3.1.1). Furthermore, Jacovi et al. [63] observed that the filters together with the max-pooling layer work as n-gram detector for tasks like sentiment classification. Understanding how the convolution layer of IWE works in the task of generating word embeddings as a prediction based model will be a challenging yet important step towards deeper insights about the model (and other deep neural network-based models in general).

5.2.2 Extensions

There are some potential directions of research that form natural continuations or extensions of our work. We discuss some of them in the following sections.

Using contextualized word embeddings One of the limitations of CBOW or IWE is that they form one representation of words that can have multiple meanings depending on the context (‘stick’, ‘bank’, ‘crane’, ‘date’ and many more). Instead of fixing a single embedding for these words, models like ELMo [106] and BERT [34] looks at the entire sentence before assigning each word an embedding—these models produce *contextualized* word embeddings. We have also observed from our experiments reported in Table 4.8 and Fig. 4.12 the importance of word ordering and context features for capturing semantic information in word embeddings. Hence, it will be interesting to see the retrieval effectiveness of QE using contextualized embeddings of models like ELMo and BERT. The models are made of complex neural network architectures like bi-directional LSTMs [49] or transformers [140] and are trained on the language modeling task (Section 2.2.3) using a large amount of data. These models can be fine-tuned on specific downstream tasks and have shown state-of-the-art performance in a number of NLP[4] [34] and IR[5] [156, 154, 110] tasks. Hence, a possible future direction is to observe whether fine-tuning the embeddings on a query-specific corpus, which are smaller with a limited vocabulary, causes an increase in retrieval effectiveness by enhancing the quality of QE terms. Perhaps, for these models, QE using the pre-trained global embeddings are sufficient for our task of passage retrieval.

Different way of obtaining QE term The error analysis of the IWE model (Section 4.3.7) and the CBOW model (Section 4.3.4) highlighted a drawback of the QE methodology employed in our work. Our QE terms are the ones that are closest to the mean of the query terms. A bad candidate term, being highly similar to one of the query terms, will make them closer to the query mean irrespective of it being not similar to other query terms. In this way, bad QE terms can be in the expanded query leading to a decrease in retrieval effectiveness. One of the possible research directions is to employ other methodologies of obtaining QE terms (rather than simply calculating the distance of the candidate terms from the query mean). For example, we can weigh each term with their IDF score in the candidate set of passages and obtain QE terms based on this weighted query mean. The intuition is to have greater influence by the more important terms in the questions for generating the QE terms. Zheng and Callan [162] proposes a query term weighting scheme which has shown to increase retrieval effectiveness in the ad-hoc search task. We can use the same, for obtaining QE terms.

Another direction of research is the classification of good and bad QE terms. Not all obtained QE terms help to increase the retrieval effectiveness—selecting only the ones that do the same might be beneficial. Some prior works [12, 100, 61] use a variety of techniques including using support vector machine, reinforcement learning or siamese networks for the classification and they have shown to improve retrieval effectiveness.

Different task and domain The success of QE using word embedding models trained on query-specific corpus encourages the research testing their effectiveness in other tasks like closed-domain QA [1], vertical search[6] [16] or enterprise search[7] [57]. These tasks typically involve retrieving information from specific domains like travel, real estate, legal, medical, etc. Documents pertaining to a specific domain can be perceived as candidate passages that are already known beforehand and when a user makes a specific query the word embeddings trained on the candidate passages of the specific domain can be used for QE. Thus the retrieval pipeline of QE using word embeddings trained on query-specific corpus employed in our thesis can also be employed in these tasks. For example, for the question 'Describe the game of football' word embeddings trained on the candidate passages from domains like 'Football' or 'Sports' can be used.

[4]See https://gluebenchmark.com/leaderboard. 6 out of top 10 models for various language understanding tasks are based on BERT embeddings.

[5]See http://www.msmarco.org/leaders.aspx. All of the top 10 models in the passage retrieval task on the MS Macro dataset is based on BERT embeddings.

[6]A vertical search engine focuses on one specific industry or type of content. For example, a travel search engine like Kayak, real estate site Trulia, etc.

[7]Enterprise search is the search within the collections of an enterprise

Bibliography

[1] Ali Mohamed Nabil Allam and Mohamed Hassan Haggag. The question answering systems: A survey. *International Journal of Research and Reviews in Information Sciences (IJRRIS)*, 2(3), 2012.

[2] Mohannad Almasri, Catherine Berrut, and Jean-Pierre Chevallet. A comparison of deep learning based query expansion with pseudo-relevance feedback and mutual information. In *European conference on information retrieval*, pages 709–715. Springer, 2016.

[3] Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval: The Concepts and Technology Behind Search*. Addison-Wesley Publishing Company, USA, 2nd edition, 2008. ISBN 9780321416919.

[4] Clinton L Ballard. Query refinement method for searching documents, November 16 1999. US Patent 5,987,457.

[5] Marco Baroni, Georgiana Dinu, and Germán Kruszewski. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 238–247, 2014.

[6] Nicholas J Belkin. Anomalous states of knowledge as a basis for information retrieval. *Canadian journal of information science*, 5(1):133–143, 1980.

[7] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3 (Feb):1137–1155, 2003.

[8] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(Feb):281–305, 2012.

[9] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.

[10] James Bradbury, Stephen Merity, Caiming Xiong, and Richard Socher. Quasi-recurrent neural networks. *arXiv preprint arXiv:1611.01576*, 2016.

[11] Jose Camacho-Collados and Mohammad Taher Pilehvar. From word to sense embeddings: A survey on vector representations of meaning. *Journal of Artificial Intelligence Research*, 63:743–788, 2018.

[12] Guihong Cao, Jian-Yun Nie, Jianfeng Gao, and Stephen Robertson. Selecting good expansion terms for pseudo-relevance feedback. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '08, pages 243–250, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-164-4. doi: 10.1145/1390334.1390377. URL `http://doi.acm.org/10.1145/1390334.1390377`.

[13] Shaosheng Cao and Wei Lu. Improving word embeddings with convolutional feature learning and subword information. In *AAAI*, pages 3144–3151, 2017.

[14] Claudio Carpineto and Giovanni Romano. A survey of automatic query expansion in information retrieval. *ACM Computing Surveys (CSUR)*, 44(1):1, 2012.

[15] Hugo Caselles-Dupré, Florian Lesaint, and Jimena Royo-Letelier. Word2vec applied to recommendation: Hyperparameters matter. In *Proceedings of the 12th ACM Conference on Recommender Systems*, pages 352–356. ACM, 2018.

[16] Michael Chau and Hsinchun Chen. Comparison of three vertical search spiders. *Computer*, 36(5):56–62, 2003.

[17] Danqi Chen. *Neural Reading Comprehension and Beyond*. PhD thesis, Stanford University, 2018.

[18] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879. Association for Computational Linguistics, 2017. doi: 10.18653/v1/P17-1171. URL `http://aclweb.org/anthology/P17-1171`.

[19] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1171. URL `https://www.aclweb.org/anthology/P17-1171`.

[20] Paul-Alexandru Chirita, Claudiu S Firan, and Wolfgang Nejdl. Personalized query expansion for the web. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 7–14. ACM, 2007.

[21] Noam Chomsky. *Syntactic structures*. Walter de Gruyter, 2002.

[22] Cyril Cleverdon. The cranfield tests on index language devices. In *Aslib proceedings*, volume 19, pages 173–194. MCB UP Ltd, 1967.

[23] Daniel Cohen and W Bruce Croft. End to end long short term memory networks for non-factoid question answering. In *Proceedings of the 2016 ACM International Conference on the Theory of Information Retrieval*, pages 143–146. ACM, 2016.

[24] Daniel Cohen, W. Bruce Croft, and Liu Yang. Wikipassageqa: A benchmark collection for research on non-factoid answer passage retrieval. In *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 165–168, 2017. ISBN: 978-1-4503-5657-2.

[25] Francesco Colace, Massimo De Santo, Luca Greco, and Paolo Napoletano. Improving relevance feedback-based query expansion by the use of a weighted word pairs approach. *Journal of the Association for Information Science and Technology*, 66(11):2223–2234, 2015.

[26] Kevyn Collins-Thompson, Jamie Callan, Egidio Terra, and Charles LA Clarke. The effect of document retrieval quality on factoid question answering performance. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 574–575. ACM, 2004.

[27] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM, 2008.

[28] W Bruce Croft and David J Harper. Using probabilistic models of document retrieval without relevance information. *Journal of documentation*, 35(4):285–295, 1979.

[29] Steve Cronen-Townsend, Yun Zhou, and W Bruce Croft. Predicting query performance. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 299–306. ACM, 2002.

[30] Ritendra Datta, Dhiraj Joshi, Jia Li, and James Z. Wang. Image retrieval: Ideas, influences, and trends of the new age. *ACM Comput. Surv.*, 40(2):5:1–5:60, May 2008. ISSN 0360-0300. doi: 10.1145/1348246.1348248. URL http://doi.acm.org/10.1145/1348246.1348248.

[31] Yann N Dauphin, Angela Fan, Michael Auli, and David Grangier. Language modeling with gated convolutional networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 933–941. JMLR.org, 2017.

[32] Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407, 1990.

[33] Mostafa Dehghani, Hosein Azarbonyad, Jaap Kamps, and Maarten de Rijke. Learning to transform, combine, and reason in open-domain question answering. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, WSDM '19, pages 681–689, New York, NY, USA, 2019. ACM. ISBN 978-1-4503-5940-5. doi: 10.1145/3289600.3291012. URL `http://doi.acm.org/10.1145/3289600.3291012`.

[34] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018.

[35] Paramveer Dhillon, Dean P Foster, and Lyle H Ungar. Multi-view learning of word embeddings via cca. In *Advances in neural information processing systems*, pages 199–207, 2011.

[36] Fernando Diaz and Donald Metzler. Improving the estimation of relevance models using large external corpora. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 154–161. ACM, 2006.

[37] Fernando Diaz, Bhaskar Mitra, and Nick Craswell. Query expansion with locally-trained word embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 367–377, Berlin, Germany, August 2016. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/P16-1035`.

[38] Efthimis N Efthimiadis. Query expansion. *Annual review of information science and technology (ARIST)*, 31:121–87, 1996.

[39] Yixing Fan, Jiafeng Guo, Yanyan Lan, Jun Xu, Chengxiang Zhai, and Xueqi Cheng. Modeling diverse relevance patterns in ad-hoc retrieval. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 375–384. ACM, 2018.

[40] Maryam Fanaeepour, Adam Makarucha, and Jey Han Lau. Evaluating word embedding hyper-parameters for similarity and analogy tasks. *arXiv preprint arXiv:1804.04211*, 2018.

[41] Hui Fang, Tao Tao, and Chengxiang Zhai. Diagnostic evaluation of information retrieval models. *ACM Trans. Inf. Syst.*, 29(2):7:1–7:42, April 2011. ISSN 1046-8188. doi: 10.1145/1961209.1961210. URL `http://doi.acm.org/10.1145/1961209.1961210`.

[42] John R Firth. A synopsis of linguistic theory, 1930-1955. *Studies in linguistic analysis*, 1957.

[43] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 36(4):193–202, 1980.

[44] Robert Gaizauskas and Kevin Humphreys. A combined ir/nlp approach to question answering against large text collections. In *Content-Based Multimedia Information Access-Volume 2*, pages 1288–1304. LE CENTRE DE HAUTES ETUDES INTERNATIONALES D'INFORMATIQUE DOCUMENTAIRE, 2000.

[45] Yoav Goldberg. A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research*, 57:345–420, 2016.

[46] Gene H Golub and Christian Reinsch. Singular value decomposition and least squares solutions. *Numerische mathematik*, 14(5):403–420, 1970.

[47] Chengyue Gong, Di He, Xu Tan, Tao Qin, Liwei Wang, and Tie-Yan Liu. Frage: frequency-agnostic word representation. In *Advances in Neural Information Processing Systems*, pages 1334–1345, 2018.

[48] Joshua Goodman. Classes for fast maximum entropy training. page 561â564, 2001.

[49] Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks*, 18 (5-6):602–610, 2005.

[50] Jiafeng Guo, Yixing Fan, Qingyao Ai, and W. Bruce Croft. A deep relevance matching model for ad-hoc retrieval. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, CIKM '16, pages 55–64, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4073-1. doi: 10.1145/2983323.2983769. URL http://doi.acm.org/10.1145/2983323.2983769.

[51] Jiafeng Guo, Yixing Fan, Liang Pang, Liu Yang, Qingyao Ai, Hamed Zamani, Chen Wu, W Bruce Croft, and Xueqi Cheng. A deep look into neural ranking models for information retrieval. *arXiv preprint arXiv:1903.06902*, 2019.

[52] Matthias Hagen, Michael Völske, Steve Göring, and Benno Stein. Axiomatic result re-ranking. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pages 721–730. ACM, 2016.

[53] P Hancox. A brief history of natural language processing, 2011.

[54] Donna Harman. Relevance feedback and other query modification techniques., 1992.

[55] Zellig S Harris. Distributional structure. *Word*, 10(2-3):146–162, 1954.

[56] Helia Hashemi, Mohammad Aliannejadi, Hamed Zamani, and W Bruce Croft. Antique: A non-factoid question answering benchmark. *arXiv preprint arXiv:1905.08957*, 2019.

[57] David Hawking. Challenges in enterprise search. In *Proceedings of the 15th Australasian database conference-Volume 27*, pages 15–24. Australian Computer Society, Inc., 2004.

[58] Johannes Hellrich and Udo Hahn. Bad companyâneighborhoods in neural embedding spaces considered harmful. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2785–2796, 2016.

[59] Harold Hotelling. Canonical correlation analysis (cca). *Journal of Educational Psychology*, page 10, 1935.

[60] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pages 2333–2338. ACM, 2013.

[61] Ayyoob Imani, Amir Vakili, Ali Montazer, and Azadeh Shakery. Deep neural networks for query expansion using word embeddings. *arXiv preprint arXiv:1811.03514*, 2018.

[62] KSD Ishwari, AKRR Aneeze, S Sudheesan, HJDA Karunaratne, A Nugaliyadde, and Y Mallawarrachchi. Advances in natural language question answering: A review. *arXiv preprint arXiv:1904.05276*, 2019.

[63] Alon Jacovi, Oren Sar Shalom, and Yoav Goldberg. Understanding convolutional neural networks for text classification. *arXiv preprint arXiv:1809.08037*, 2018.

[64] Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. Character-aware neural language models. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[65] Lorena Kodra and Elinda Kajo Meçe. Question answering systems: A review on present developments, challenges and trends. *Department of Computer Engineering, Polytechnic University of Tirana, Albania*, 2017.

[66] Marijn Koolen, Toine Bogers, Maria Gäde, Mark Hall, Hugo Huurdeman, Jaap Kamps, Mette Skov, Elaine Toms, and David Walsh. Overview of the clef 2015 social book search lab. In *International conference of the cross-language evaluation forum for European languages*, pages 545–564. Springer, 2015.

[67] J Kupiec and A MURAX. Robust linguistic approach for question answering using an on-line encyclopedia. In *Proceedings of ACM-SIGIR*, volume 93.

[68] Saar Kuzi, Anna Shtok, and Oren Kurland. Query expansion using word embeddings. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, CIKM '16, pages 1929–1932, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4073-1. doi: 10.1145/2983323.2983876. URL http://doi.acm.org/10.1145/2983323.2983876.

[69] Tessa Lau and Eric Horvitz. Patterns of search: analyzing and modeling web query refinement. In *UM99 User Modeling*, pages 119–128. Springer, 1999.

[70] Rémi Lebret and Ronan Collobert. Word embeddings through hellinger PCA. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 482–490, Gothenburg, Sweden, April 2014. Association for Computational Linguistics. doi: 10.3115/v1/E14-1051. URL `https://www.aclweb.org/anthology/E14-1051`.

[71] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521 (7553):436, 2015.

[72] Omer Levy and Yoav Goldberg. Linguistic regularities in sparse and explicit word representations. In *Proceedings of the eighteenth conference on computational natural language learning*, pages 171–180, 2014.

[73] Omer Levy and Yoav Goldberg. Neural word embedding as implicit matrix factorization. In *Advances in neural information processing systems*, pages 2177–2185, 2014.

[74] Omer Levy, Yoav Goldberg, and Ido Dagan. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics*, 3:211–225, 2015.

[75] Yang Li and Tao Yang. Word embedding for understanding natural language: A survey. In *Guide to Big Data Applications*, pages 83–104. Springer, 2018.

[76] Dekang Lin. Automatic retrieval and clustering of similar words. In *COLING 1998 Volume 2: The 17th International Conference on Computational Linguistics*, volume 2, 1998.

[77] Tal Linzen. Issues in evaluating semantic spaces using word analogies. *arXiv preprint arXiv:1606.07736*, 2016.

[78] Kevin Lund and Curt Burgess. Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior research methods, instruments, & computers*, 28(2):203–208, 1996.

[79] Xuezhe Ma and Eduard Hovy. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1101. URL `https://www.aclweb.org/anthology/P16-1101`.

[80] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[81] David JC MacKay and Linda C Bauman Peto. A hierarchical dirichlet language model. *Natural language engineering*, 1(3):289–308, 1995.

[82] Christopher Manning, Prabhakar Raghavan, and Hinrich Schütze. Introduction to information retrieval. *Natural Language Engineering*, 16(1):100–103, 2010.

[83] Lilyana Mihalkova and Raymond Mooney. Learning to disambiguate search queries from short sessions. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 111–127. Springer, 2009.

[84] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Eleventh annual conference of the international speech communication association*, 2010.

[85] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013. URL `http://arxiv.org/abs/1301.3781`.

[86] Tomas Mikolov, Quoc V Le, and Ilya Sutskever. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*, 2013.

[87] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pages 3111–3119, 2013. URL `http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality`

[88] Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751. Association for Computational Linguistics, 2013. URL `http://aclweb.org/anthology/N13-1090`.

[89] Amit Mishra and Sanjay Kumar Jain. A survey on question answering systems with classification. *Journal of King Saud University-Computer and Information Sciences*, 28(3):345–361, 2016.

[90] Bhaskar Mitra and Nick Craswell. Neural models for information retrieval, May 2017. URL `https://www.microsoft.com/en-us/research/publication/neural-models-information-retrieval/`.

[91] Bhaskar Mitra, Fernando Diaz, and Nick Craswell. Learning to match using local and distributed representations of text for web search. In *Proceedings of the 26th International Conference on World Wide Web*, pages 1291–1299. International World Wide Web Conferences Steering Committee, 2017.

[92] Bhaskar Mitra, Nick Craswell, et al. An introduction to neural information retrieval. *Foundations and Trends® in Information Retrieval*, 13(1):1–126, 2018.

[93] Andriy Mnih and Geoffrey Hinton. Three new graphical models for statistical language modelling. In *Proceedings of the 24th international conference on Machine learning*, pages 641–648. ACM, 2007.

[94] Andriy Mnih and Geoffrey E Hinton. A scalable hierarchical distributed language model. In *Advances in neural information processing systems*, pages 1081–1088, 2009.

[95] Frederic Morin and Yoshua Bengio. Hierarchical probabilistic neural network language model. In *Aistats*, volume 5, pages 246–252. Citeseer, 2005.

[96] Eric Nalisnick, Bhaskar Mitra, Nick Craswell, and Rich Caruana. Improving document ranking with dual word embeddings. In *Proceedings of the 25th International Conference Companion on World Wide Web*, WWW '16 Companion, pages 83–84, Republic and Canton of Geneva, Switzerland, 2016. International World Wide Web Conferences Steering Committee. ISBN 978-1-4503-4144-8. doi: 10.1145/2872518.2889361. URL https://doi.org/10.1145/2872518.2889361.

[97] Federico Nanni, Bhaskar Mitra, Matt Magnusson, and Laura Dietz. Benchmark for complex answer retrieval. In *Proceedings of the ACM SIGIR International Conference on Theory of Information Retrieval*, ICTIR '17, pages 293–296, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-4490-6. doi: 10.1145/3121050.3121099. URL http://doi.acm.org/10.1145/3121050.3121099.

[98] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*, 2016.

[99] Malvina Nissim, Rik van Noord, and Rob van der Goot. Fair is better than sensational: Man is to doctor as woman is to doctor. *arXiv preprint arXiv:1905.09866*, 2019.

[100] Rodrigo Nogueira and Kyunghyun Cho. Task-oriented query reformulation with reinforcement learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 574–583. Association for Computational Linguistics, 2017. doi: 10.18653/v1/D17-1061. URL http://aclweb.org/anthology/D17-1061.

[101] Nawal Ould-Amer, Philippe Mulhem, and Mathias Gery. Toward word embedding for personalized information retrieval, 2016.

[102] Dipasree Pal, Mandar Mitra, and Kalyankumar Datta. Improving query expansion using wordnet. *Journal of the Association for Information Science and Technology*, 65(12):2469–2478, 2014.

[103] Hamid Palangi, Li Deng, Yelong Shen, Jianfeng Gao, Xiaodong He, Jianshu Chen, Xinying Song, and R Ward. Semantic modelling with long-short-term memory for information retrieval. *arXiv preprint arXiv:1412.6629*, 2014.

[104] Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Jingfang Xu, and Xueqi Cheng. Deeprank: A new deep architecture for relevance ranking in information retrieval. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 257–266. ACM, 2017.

[105] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[106] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237. Association for Computational Linguistics, 2018. doi: 10.18653/v1/N18-1202. URL http://aclweb.org/anthology/N18-1202.

[107] Ngoc-Quan Pham, German Kruszewski, and Gemma Boleda. Convolutional neural network language models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1153–1162, 2016.

[108] Benedicte Pierrejean and Ludovic Tanguy. Predicting word embeddings variability. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 154–159, 2018.

[109] Jay Michael Ponte and W Bruce Croft. *A language modeling approach to information retrieval*. PhD thesis, University of Massachusetts at Amherst, 1998.

[110] Yifan Qiao, Chenyan Xiong, Zhenghao Liu, and Zhiyuan Liu. Understanding the behaviors of bert in ranking. *arXiv preprint arXiv:1904.07531*, 2019.

[111] Deepak Ravichandran and Eduard Hovy. Learning surface text patterns for a question answering system. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 41–47. Association for Computational Linguistics, 2002.

[112] Daniël Rennings, Felipe Moraes, and Claudia Hauff. An axiomatic approach to diagnosing neural ir models. In *European Conference on Information Retrieval*, pages 489–503. Springer, 2019.

[113] Michael Repplinger, Lisa Beinborn, and Willem Zuidema. Vector-space models of words and sentences.

[114] C. J. Van Rijsbergen. *Information Retrieval*. Butterworth-Heinemann, Newton, MA, USA, 2nd edition, 1979. ISBN 0408709294.

[115] Stephen E Robertson. The probability ranking principle in ir. *Journal of documentation*, 33(4):294–304, 1977.

[116] Joseph John Rocchio. Relevance feedback in information retrieval. *The SMART retrieval system: experiments in automatic document processing*, pages 313–323, 1971.

[117] Douglas LT Rohde, Laura M Gonnerman, and David C Plaut. An improved model of semantic similarity based on lexical co-occurrence. *Communications of the ACM*, 8(627-633):116, 2006.

[118] Dwaipayan Roy, Debjyoti Paul, Mandar Mitra, and Utpal Garain. Using word embeddings for automatic query expansion. In *Neu-IR: The SIGIR 2016 Workshop on Neural Information Retrieval*, 2016.

[119] Gerard Salton and Chris Buckley. Improving retrieval performance by relevance feedback. *Journal of the American society for information science*, 41(4):288–297, 1990.

[120] Gerard Salton, Anita Wong, and Chung-Shu Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975.

[121] Tefko Saracevic. Relevance: A review of the literature and a framework for thinking on the notion in information science. part iii: Behavior and effects of relevance. *Journal of the American Society for information Science and Technology*, 58(13):2126–2144, 2007.

[122] Ruhi Sarikaya, Paul A Crook, Alex Marin, Minwoo Jeong, Jean-Philippe Robichaud, Asli Celikyilmaz, Young-Bum Kim, Alexandre Rochette, Omar Zia Khan, Xiaohu Liu, et al. An overview of end-to-end language understanding and dialog management for personal digital assistants. In *2016 IEEE Spoken Language Technology Workshop (SLT)*, pages 391–397. IEEE, 2016.

[123] Roger C Schank. Conceptual dependency: A theory of natural language understanding. *Cognitive psychology*, 3(4):552–631, 1972.

[124] Aliaksei Severyn and Alessandro Moschitti. Learning to rank short text pairs with convolutional deep neural networks. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pages 373–382. ACM, 2015.

[125] Saeedeh Shekarpour, Edgard Marx, Sören Auer, and Amit Sheth. Rquery: rewriting natural language queries on knowledge graphs to alleviate the vocabulary mismatch problem. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[126] Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. A latent semantic model with convolutional-pooling structure for information retrieval. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 101–110. ACM, 2014.

[127] Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. Learning semantic representations using convolutional neural networks for web search. In *Proceedings of the 23rd International Conference on World Wide Web*, pages 373–374. ACM, 2014.

[128] Scharolta Katharina Sienčnik. Adapting word2vec to named entity recognition. In *Proceedings of the 20th nordic conference of computational linguistics, nodalida 2015, may 11-13, 2015, vilnius, lithuania*, number 109, pages 239–243. Linköping University Electronic Press, 2015.

[129] Robert F Simmons and John F Burger. A semantic analyzer for english sentences. *Mech. Translat. & Comp. Linguistics*, 11(1-2):1–13, 1968.

[130] Robert F Simmons, Sheldon Klein, and Keren McConlogue. Indexing and dependency logic for answering english questions. *American Documentation*, 15 (3):196–204, 1964.

[131] K Spark-Jones. Report on the need for and provision of an'ideal'information retrieval test collection. *Computer Laboratory*, 1975.

[132] Amanda Spink, Dietmar Wolfram, Major BJ Jansen, and Tefko Saracevic. Searching the web: The public and their queries. *Journal of the American society for information science and technology*, 52(3):226–234, 2001.

[133] Trevor Strohman, Donald Metzler, Howard Turtle, and W. B. Croft. Indri: A language model-based search engine for complex queries. In *Proceedings of the International Conference on Intelligence Analysis*, 2004.

[134] Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Kathryn Mazaitis, Ruslan Salakhutdinov, and William Cohen. Open domain question answering using early fusion of knowledge bases and text. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4231–4242, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D18-1455.

[135] Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. Lstm neural networks for language modeling. In *Thirteenth annual conference of the international speech communication association*, 2012.

[136] Mihai Surdeanu, Massimiliano Ciaramita, and Hugo Zaragoza. Learning to rank answers to non-factoid questions from web collections. *Computational linguistics*, 37(2):351–383, 2011.

[137] Zhiwen Tang and Grace Hui Yang. Deeptilebars: Visualizing term distribution for neural information retrieval. *CoRR*, abs/1811.00606, 2018. URL http://arxiv.org/abs/1811.00606.

[138] Simone Teufel. *An Overview of Evaluation Methods in TREC Ad Hoc Information Retrieval and TREC Question Answering*, pages 163–186. Springer Netherlands, Dordrecht, 2007. ISBN 978-1-4020-5817-2. doi: 10.1007/978-1-4020-5817-2_6. URL https://doi.org/10.1007/978-1-4020-5817-2_6.

[139] Peter D Turney and Patrick Pantel. From frequency to meaning: Vector space models of semantics. *Journal of artificial intelligence research*, 37:141–188, 2010.

[140] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

[141] Lakshmi Vikraman, W Bruce Croft, and Brendan O'Connor. Exploring diversification in non-factoid question answering. In *Proceedings of the 2018 ACM SIGIR International Conference on Theory of Information Retrieval*, pages 223–226. ACM, 2018.

[142] Ellen M Voorhees. The philosophy of information retrieval evaluation. In *Workshop of the cross-language evaluation forum for european languages*, pages 355–370. Springer, 2001.

[143] Ellen M Voorhees and Donna K Harman. Overview of the seventh text retrieval conference (trec-7).

[144] Ellen M Voorhees et al. The trec-8 question answering track report. In *Trec*, volume 99, pages 77–82. Citeseer, 1999.

[145] Shengxian Wan, Yanyan Lan, Jiafeng Guo, Jun Xu, Liang Pang, and Xueqi Cheng. A deep architecture for semantic matching with multiple positional sentence representations. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[146] Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerry Tesauro, Bowen Zhou, and Jing Jiang. R3: Reinforced ranker-reader for open-domain question answering, 2018. URL `https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16712`.

[147] William A Woods. Augmented transition networks for natural language analysis. 1969.

[148] Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russell Power. End-to-end neural ad-hoc ranking with kernel pooling. In *Proceedings of the 40th International ACM SIGIR conference on research and development in information retrieval*, pages 55–64. ACM, 2017.

[149] Bo Xu, Hongfei Lin, Yuan Lin, Liang Yang, and Kan Xu. Improving pseudo-relevance feedback with neural network-based word representations. *IEEE Access*, 6:62152–62165, 2018.

[150] Yang Xu, Gareth J.F. Jones, and Bin Wang. Query dependent pseudo-relevance feedback based on wikipedia. In *Proceedings of the 32Nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '09, pages 59–66, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-483-6. doi: 10.1145/1571941.1571954. URL `http://doi.acm.org/10.1145/1571941.1571954`.

[151] Bai Xue, Chen Fu, and Zhan Shaobin. A study on sentiment computing and classification of sina weibo with word2vec. In *2014 IEEE International Congress on Big Data*, pages 358–363. IEEE, 2014.

[152] Liu Yang, Qingyao Ai, Jiafeng Guo, and W Bruce Croft. anmm: Ranking short answer texts with attention-based neural matching model. In *Proceedings*

of the 25th ACM International on Conference on Information and Knowledge Management, pages 287–296. ACM, 2016.

[153] Liu Yang, Qingyao Ai, Damiano Spina, Ruey-Cheng Chen, Liang Pang, W Bruce Croft, Jiafeng Guo, and Falk Scholer. Beyond factoid qa: effective methods for non-factoid answer sentence retrieval. In *European Conference on Information Retrieval*, pages 115–128. Springer, 2016.

[154] Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. End-to-end open-domain question answering with BERTserini. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 72–77, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/N19-4013`.

[155] Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. End-to-end open-domain question answering with BERTserini. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 72–77, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/N19-4013`.

[156] Wei Yang, Haotian Zhang, and Jimmy Lin. Simple applications of bert for ad hoc document retrieval. *arXiv preprint arXiv:1903.10972*, 2019.

[157] Xing Yi and James Allan. Indri at trec 2007: Million query (1mq) track. In *TREC*, 2007.

[158] Wen-tau Yih and Hao Ma. Question answering with knowledge base, web and beyond. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 1219–1221. ACM, 2016.

[159] Hamed Zamani and W. Bruce Croft. Embedding-based query language models. In *Proceedings of the 2016 ACM International Conference on the Theory of Information Retrieval*, ICTIR '16, pages 147–156, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4497-5. doi: 10.1145/2970398.2970405. URL `http://doi.acm.org/10.1145/2970398.2970405`.

[160] Chengxiang Zhai and John Lafferty. Model-based feedback in the language modeling approach to information retrieval. In *Proceedings of the tenth international conference on Information and knowledge management*, pages 403–410. ACM, 2001.

[161] Le Zhao and Jamie Callan. Automatic term mismatch diagnosis for selective query expansion. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pages 515–524. ACM, 2012.

[162] Guoqing Zheng and Jamie Callan. Learning to reweight terms with distributed representations. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pages 575–584. ACM, 2015.

[163] Dong Zhou, Séamus Lawless, and Vincent Wade. Improving search via personalized query expansion using social media. *Information retrieval*, 15(3-4): 218–242, 2012.

[164] Liron Zighelnic and Oren Kurland. Query-drift prevention for robust query expansion. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 825–826. ACM, 2008.

Appendix A

Convolution in Text

Fig. A.1 shows how convolutional layer forms the context representation in details. The input to the convolutional layer is the context words represented using their character-trigrams and root affixes. There are also zero padding on both sides of the context which means the number of outputs from the convolution operation is equal to the number of context words which, in this example, is 4. We require the context embedding to be of the same dimensions (d) as the word embeddings. Hence, we need d filters to learn the context representation. For our experiments (and in this example), we use a sliding windoe l of 3. Each filter (f_1, $f_2, ..., f_d$) slides over the input representations and forms 4 outputs each with the convolution operation $< [\mathbf{u}_i, ..., \mathbf{u}_{i+l-1}], f_j >$. The filters are represented using different colours in Fig. A.1. Thus, from the convolution operation we obtain a $d \times 4$ output. After that in the max pooling operation, we take the maximum value obtained using each filter. That is we take the maximum out of the four outputs from $< [\mathbf{u}_i, ..., \mathbf{u}_{i+l-1}], f_1 >$ by filter f_1. We repeat this for all d filters and finally we get the $d \times 1$ representation of the context or the context embedding $\mathbf{c}$.

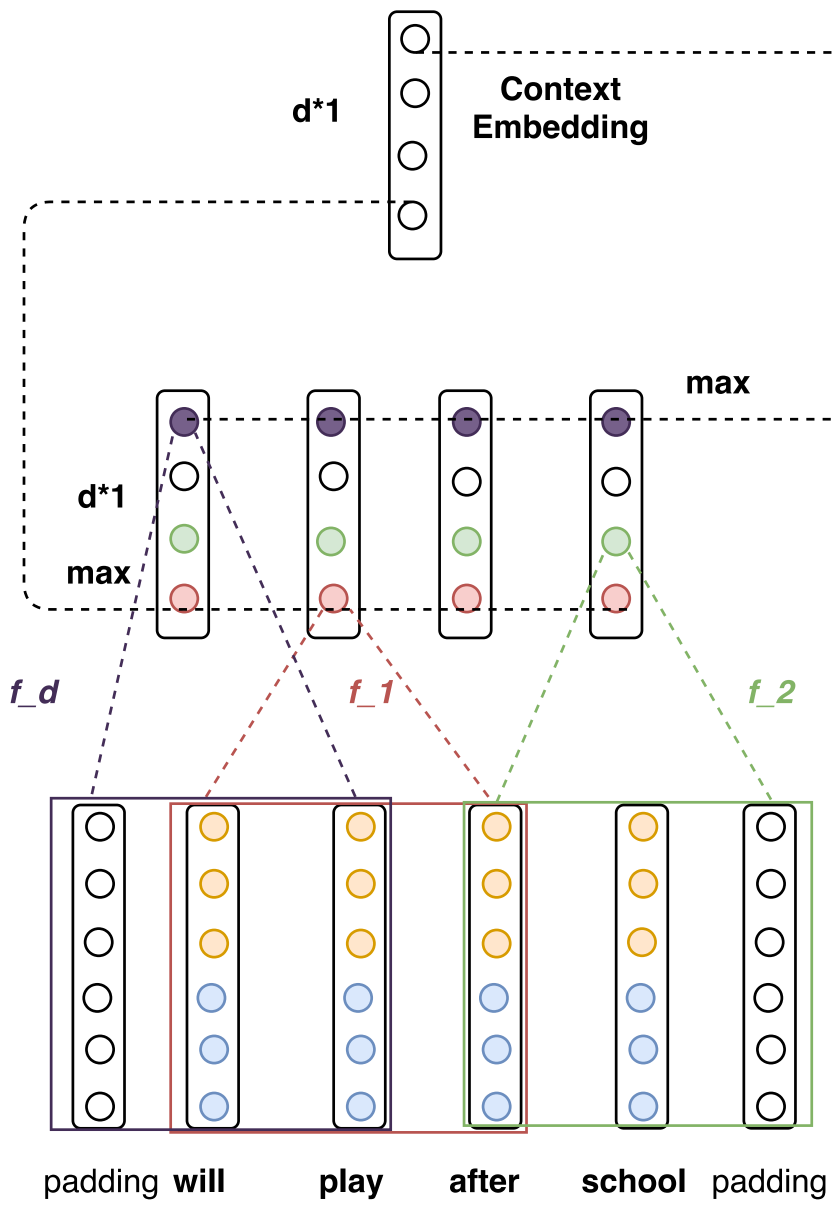


Figure A.1: A detailed schematic of the convolutional feature learning of the context.

Appendix B

Hyperparameter Tuning

This is the overview of hyperparameter values tested while tuning the performance of traditional IR models QL, RM3 and RM3+EC and word embedding models CBOW and IWE.

Model	Hyperparameters	Tested Values
QL	μ	{5, 10, 100, 250, 500, 750, 1000, 2000, 3000, 4000, 5000}
RM3	$fbMu$	{250, 500, 750, 1000, 3000}
	$fbDocs$	{5, 10, 15, 20, 25, 30, 40, 50, 60, 70, 80}
	$fbTerms$	{5, 100, 200, 350, 400, 450, 500}
	$fbOrigWeight$	{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}
	μ	{5, 10, 100, 250, 500, 750, 1000, 2000, 3000, 4000, 5000}
RM3 + EC	$fbMu$	{250, 500, 750, 1000, 3000}
	$fbDocs$	{5, 10, 50, 100, 150, 200, 250, 350, 400, 450, 500}
	$fbTerms$	{5, 100, 200, 350, 400, 450, 500}
	$fbOrigWeight$	{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}
	μ	{5, 10, 100, 250, 500, 750, 1000, 2000, 3000, 4000, 5000}
CBOW	α	{0.1, 0.01, 0.001}
	d	{50, 100, 200, 300, 400}
	k	{5, 50, 100, 200, 300, 400, 500}
	λ	{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}
	μ_2	{100, 250, 500, 750, 1000, 3000, 5000}
IWE	d	{50, 100, 200, 300, 400}
	k	{5, 50, 100, 200, 300, 400, 500}
	λ	{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}
	μ	{100, 250, 500, 750, 1000, 3000, 5000}

Table B.1: Range of hyperparameter values tested while tuning the MAP of the IR models. They are tuned on the train and dev split of `WikiPassageQA` dataset.

Appendix C

Long Questions

As we have seen previously, **complex inference** type error was one of the most prevalent errors for QE using most of the word embedding models. Moreover, for the IWE model 40% of the improvements over the CBOW model were achieved in the **complex inference** error type. The **complex inference** type errors were typically committed on queries which were long and needed long distance semantic understanding of question terms. We can also observe from Fig. 4.7 that the IWE model performs better than all other models, specially the CBOW local target model, for longer queries.

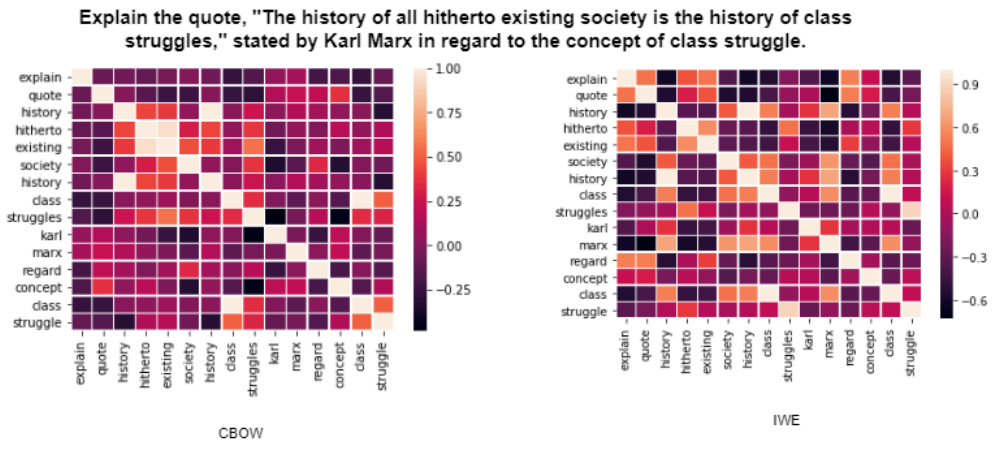


Figure C.1: Correlation of distances from generated expansion terms among the terms in a query. A high correlation among the terms indicate that the expansion terms generated were similar to the words. Ideally, we would like to have high correlation among words which are far away from each other in long questions to have expansion terms similar to query terms which are syntactically far apart but equally important semantically.

Now a long query has multiple parts which contributes to the entire semantic meaning of the question. For example, ‘Explain the quote, "The history of all hitherto existing society is the history of class struggles," stated by Karl Marx in regard to the concept of class struggle.’ - this question contains three main components - the quote, ‘Karl Marx’ and ‘concept of class struggle’. The query expansion model should ideally find expansion terms that is similar to all these components. In this experiment, we are trying to observe the difference between IWE and CBOW models trained on local target data while dealing with longer queries.

For this experiment, we find the cosine similarity of the QE terms for the above question generated by the embeddings of the two models with each of the terms in the query. In Fig. C.1 we plot the correlation of these distances among the query terms. The intuition is that query terms having high correlation contributes to the generation of similar expansion terms. We would want the important parts in the question to be correlated to each other which would mean that the generated expansion terms have similar distance to the correlated query terms in the embedding space. If we see the CBOW model, there is a local correlation among the words in the quote and words like ‘class struggle’. There is a lack of correlation among the three different parts of the question. This means that the expansion terms generated are similar only to these local group of words and not so much to other important parts of the question like ‘Karl Marx’. Hence, these QE terms result in a decrease in retrieval effectiveness. But in the IWE model, we see high correlation among different words from all important parts of the question. For example, the word ‘Marx’ has high correlation with words from the quotes as well the word ‘class’ from ‘concept of class struggle’. Hence, we can say that the QE terms generated in this case are similar to words from all important parts of the question. Naturally, these QE terms lead to an improved retrieval effectiveness. One more interesting thing to observe is the high correlation between the word ‘struggle’ and ‘struggles’ for the IWE model. The sub-word feature information in the IWE model creates similar embeddings for words having high overlap in sub-word structure which also contributes to the generation of better QE terms.